iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** XI **Month of publication:** November 2025

DOI: https://doi.org/10.22214/ijraset.2025.75412

www.ijraset.com

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

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue XI Nov 2025- Available at www.ijraset.com

IoT Enabled Sanitation Management System

Aamaal Zahrah
Hyderabad Institute of Technology and Management, India

Abstract: ***The IoT and GSM Enabled Sanitation Management System is designed to automate and enhance sanitation monitoring using modern embedded technology. The system utilizes an ESP32 microcontroller as its central control unit, interfaced with multiple sensors such as gas, IR, and ultrasonic sensors. These sensors continuously monitor environmental conditions such as gas leakage, human detection, and waste bin level. When abnormal conditions are detected, the system activates appropriate actuators like buzzers or pumps and sends alerts via GSM and IoT modules. The system operates on solar power, ensuring sustainability and continuous functionality even in remote areas. Real-time data visualization is provided through an LCD display and cloud connectivity, enabling remote monitoring and management of sanitation conditions.***

I. INTRODUCTION

Sanitation management plays a crucial role in maintaining public health and hygiene, particularly in urban and rural waste management systems. Traditional sanitation monitoring methods are often labor-intensive, inefficient, and unable to provide real-time updates. To overcome these limitations, this project proposes an IoT and GSM-based sanitation management system utilizing the ESP32 microcontroller. The proposed system integrates multiple sensors for environmental and waste monitoring. The gas sensor detects harmful gases or odors, activating an AC pump or alert system if gas concentration exceeds safe limits. The IR sensor detects human presence to automate cleaning or pumping operations, while the ultrasonic sensor measures waste bin levels for overflow prevention. The GSM 800C module enables SMS-based alerts, and the IoT module uploads real-time data to a cloud server for remote access. The system is powered by a solar-powered battery, ensuring ecofriendly and uninterrupted operation. Additionally, an LCD provides on-site display of status information, and a buzzer acts as an immediate warning system. Through automation and remote connectivity, this project enhances the efficiency, safety, and reliability of sanitation management in public and private spaces.

A. Introduction of Embedded System

An Embedded System is a combination of computer hardware and software, and perhaps additional mechanical or other parts, designed to perform a specific function. A good example is the microwave oven. Almost every household has one, and tens of millions of them are used every day, but very few people realize that a processor and software are involved in the preparation of their lunch or dinner.

This is in direct contrast to the personal computer in the family room. It too is comprised of computer hardware and software and mechanical components (disk drives, for example). However, a personal computer is not designed to perform a specific function rather; it is able to do many different things. Many people use the term general-purpose computer to make this distinction clear. As shipped, a general-purpose computer is a blank slate; the manufacturer does not know what the customer will do wish it. One customer may use it for a network file server another may use it exclusively for playing games, and a third may use it to write the next great American novel.

Frequently, an embedded system is a component within some larger system. For example, modern cars and trucks contain many embedded systems. One embedded system controls the anti-lock brakes, other monitors and controls the vehicle's emissions, and a third displays information on the dashboard. In some cases, these embedded systems are connected by some sort of a communication network, but that is certainly not a requirement.

At the possible risk of confusing you, it is important to point out that a general-purpose computer is itself made up of numerous embedded systems. For example, my computer consists of a keyboard, mouse, video card, modem, hard drive, floppy drive, and sound card-each of Which is an embedded system? Each of these devices contains a processor and software and is designed to perform a specific function. For example, the modem is designed to send and receive digital data over analog telephone line. That's it and all of the other devices can be summarized in a single sentence as well.

If an embedded system is designed well, the existence of the processor and software could be completely unnoticed by the user of the device. Such is the case for a microwave oven, VCR, or alarm clock.

In some cases, it would even be possible to build an equivalent device that does not contain the processor and software. This could be done by replacing the combination with a custom integrated circuit that performs the same functions in hardware. However, a lot of flexibility is lost when a design is hard-cooled in this way. It is mush easier, and cheaper, to change a few lines of software than to redesign a piece of custom hardware.

B. History and Future

Given the definition of embedded systems earlier is this chapter; the first such systems could not possibly have appeared before 1971. That was the year Intel introduced the world's first microprocessor. This chip, the 4004, was designed for use in a line of business calculators produced by the Japanese Company Busicom. In 1969, Busicom asked Intel to design a set of custom integrated circuits-one for each of their new calculator models. The 4004 was Intel's response rather than design custom hardware for each calculator, Intel proposed a general-purpose circuit that could be used throughout the entire line of calculators. Intel's idea was that the software would give each calculator its unique set of features.

The microcontroller was an overnig"t su'cess, and its use for ancreased steadily over the next decade. Early embedded applications included unmanned space probes, computerized traffic lights, and aircraft flight control systems. In the 1980s, embedded systems quietly rode the waves of the microcomputer age and brought microprocessors into every part of our kitchens (bread machines, food processors, and microwave ovens), living rooms (televisions, stereos, and remote controls), and workplaces (fax machines, pagers, laser printers, cash registers, and credit card readers).

It seems inevitable hat the number of embedded systems will continue to increase rapidly. Already there are promising new embedded devices that have enormous market potential; light switches and thermostats that can be central computer, intelligent air-bag systems that don't inflate when children or small adults are present, pal-sized electronic organizers and personal digital assistants (PDAs), digital cameras, and dashboard navigation systems. Clearly, individuals who possess the skills and desire to design the next generation of embedded systems will be in demand for quite some time.

C. Real Time Systems

One subclass of embedded is worthy of an introduction at this point. As commonly defined, a realtime system is a computer system that has timing constraints. In other words, a real-time system is partly specified in terms of its ability to make certain calculations or decisions in a timely manner. These important calculations are said to have deadlines for completion. And, for all practical purposes, a missed deadline is just as bad as a wrong answer.

The issue of what if a deadline for as missed for as a crucial one. For example, for af the real-time system is part of an airplane's flight control system, it is possible for the lives of the passengers and crew to be endangered by a single missed deadline. However, if instead the system is involved in satellite communication, the damage could be limited to a single corrupt data packet. The more severe the consequences, the more likely it will be said that the deadline is "hard" and thus, the system is a hard real-time system. Real-time systems at the other end of this discussion are said to have "soft" deadlines.

All of the topics and examples presented in this book are applicable to the designers of real-time system who is more delight in his work. He must guarantee reliable operation of the software and hardware under all the possible conditions and to the degree that human lives depend upon three system's proper execution, engineering calculations and descriptive paperwork.

1) Application Areas :Nearly 99 per cent of the processors manufactured end up in embedded systems. The embedded system market is one of the highest growth areas as these systems are used in very market segment- consumer electronics, office automation, industrial automation, biomedical engineering, wireless communication,Data communication, telecommunications, transportation, military and so on.

2) Consumer appliances: At home we use a number of embedded systems which include digital camera, digital diary, DVD player, electronic toys, microwave oven, remote controls for TV and airconditioner, VCO player, video game consoles, video recorders etc. Today's high-tech car has about 20 embedded systems for transmission control, engine spark control, air-conditioning, navigation etc. Even wristwatches are now becoming embedded systems. The palmtops are powerful embedded systems using which we can carry out many general-purpose tasks such as playing games and word processing.

3) Office automation: The office automation products using em embedded systems are copying machine, fax machine, key telephone, modem, printer, scanner etc.

4) Industrial automation: Today a lot of industries use embedded systems for process control. These include pharmaceutical, cement, sugar, oil exploration, nuclear energy, electricity generation and transmission. The embedded systems for industrial use are designed to carry out specific tasks such as monitoring the temperature, pressure, humidity, voltage, current etc., and then

take appropriate action based on the monitored levels to control other devices or to send information to a centralized monitoring station. In hazardous industrial environment, where human presence has to be avoided, robots are used, which are programmed to do specific jobs. The robots are now becoming very powerful and carry out many interesting and complicated tasks such as hardware assembly.

5) Medical electronics: Almost every medical equipment in the hospital is an embedded system. These equipments include diagnostic aids such as ECG, EEG, blood pressure measuring devices, X-ray scanners; equipment used in blood analysis, radiation, colonoscopy, endoscopy etc. Developments in medical electronics have paved way for more accurate diagnosis of diseases.

6) Computer networking: Computer networking products such as bridges, routers, Integrated Services Digital Networks (ISDN), Asynchronous Transfer Mode (ATM), X.25 and frame relay switches are embedded systems which implement the necessary data communication protocols. For example, a router interconnects two networks. The two networks may be running different protocol stacks. The router's function is to obtain the data packets from incoming pores, analyze the packets and send them towards the destination after doing necessary protocol conversion. Most networking equipments, other than the end systems (desktop computers) we use to access the networks, are embedded systems.

7) Telecommunications: In the field of telecommunications, the embedded systems can be categorized as subscriber terminals and network equipment. The subscriber terminals such as key telephones, ISDN phones, terminal adapters, web cameras are embedded systems. The network equipment includes multiplexers, multiple access systems, Packet Assemblers Dissemblers (PADs), sate11ite modems etc. IP phone, IP gateway, IP gatekeeper etc. are the latest embedded systems that provide very low-cost voice communication over the Internet.

8) Wireless technologies: Advances in mobile communications are paving way for many interesting applications using embedded systems. The mobile phone is one of the marvels of the last decade of the 20'h century. It is a very powerful embedded system that provides voice communication while we are on the move. The Personal Digital Assistants and the palmtops can now be used to access multimedia services over the Internet. Mobile communication infrastructure such as base station controllers, mobile switching centers are also powerful embedded systems.

9) Insemination:Testing and measurement are the fundamental requirements in all scientific and engineering activities. The measuring equipment we use in laboratories to measure parameters such as weight, temperature, pressure, humidity, voltage, current etc. are all embedded systems. Test equipment such as oscilloscope, spectrum analyzer, logic analyzer, protocol analyzer, radio communication test set etc. are embedded systems built around powerful processors. Thank to miniaturization, the test and measuring equipment are now becoming portable facilitating easy testing and measurement in the field by field-personnel.

10) Security: Security of persons and information has always been a major issue. We need to protect our homes and offices; and also the information we transmit and store. Developing embedded systems for security applications is one of the most lucrative businesses nowadays. Security devices at homes, offices, airports etc. for authentication and verification are embedded systems. Encryption' devices are nearly 99 per cent of the processors that are manufactured end up in embedded systems. Embedded systems find applications in. every industrial segment- consumer electronics, transportation, avionics, biomedical engineering, manufacturing, process control and industrial automation, data communication, telecommunication, defense, security etc. Used to encrypt the data/voice being transmitted on communication links such as telephone lines. Biometric systems using fingerprint and face recognition are now being extensively used for user authentication in banking applications as well as for access control in high security buildings.

11) Finance: Financial dealing through cash and cheques are now slowly paving way for transactions using smart cards and ATM (Automatic Teller Machine, also expanded as Any Time Money) machines. Smart card, of the size of a credit card, has a small micro-controller and memory; and it interacts with the smart card reader! ATM machine and acts as an electronic wallet. Smart card technology has the capability of ushering in a cashless society. Well, the list goes on. It is no exaggeration to say that eyes wherever you go, you can see, or at least feel, the work of an embedded system.

D. Overview of Embedded System Architecture

Every embedded system consists of custom-built hardware built around a Central Processing Unit (CPU). This hardware also contains memory chips onto which the software is loaded. The software residing on the memory chip is also called the 'firmware'. The embedded system architecture can be represented as a layered architecture as shown in Fig.

The operating system runs above the hardware, and the application software runs above the operating system. The same architecture is applicable to any computer including a desktop computer. However, there are significant differences. It is not compulsory to have an operating system in every embedded system.

For small appliances such as remote control units, air conditioners, toys etc., there is no need for an operating system and you can write only the software specific to that application. For applications involving complex processing, it is advisable to have an operating system. In such a case, you need to integrate the application software with the operating system and then transfer the entire software on to the memory chip. Once the software is transferred to the memory chip, the software will continue to run fora long time you don't need to reload new software.

Now, let us see the details of the various building blocks of the hardware of an embedded system. As shown in Fig. the building blocks are:

1) Central Processing Unit (CPU): The Central Processing Unit (processor, in short) can be any of the following: microcontroller, microprocessor or Digital Signal Processor (DSP). A micro-controller is a low-cost processor. Its main attraction is that on the chip itself, there will be many other components such as memory, serial communication interface, analog-to digital converter etc. So, for small applications, a microcontroller is the best choice as the number of external components required will be very less. On the other hand, microprocessors are more powerful, but you need to use many external components with them. D5P is used mainly for applications in which signal processing is involved such as audio and video processing.

2) Memory: The memory is categorized as Random Access 11emory (RAM) and Read Only Memory (ROM). The contents of the RAM will be erased if power is switched off to the chip, whereas ROM retains the contents even if the power is switched off. So, the firmware is stored in the ROM. When power is switched on, the processor reads the ROM; the program is program is executed.

3) Input devices: Unlike the desktops, the input devices to an embedded system have very limited capability. There will be no keyboard or a mouse, and hence interacting with the embedded system is no easy task. Many embedded systems will have a small keypad-you press one key to give a specific command. A keypad may be used to input only the digits. Many embedded systems used in process control do not have any input device foruser interaction; they take inputs fromsensors or transducers 1'fnd produce electrical signals that are in turn fed to other systems.

4) Output devices: The output devices of the embedded systems also have very limited capability. Some embedded systems will have a few Light Emitting Diodes (LEDs) to indicate the health status of the system modules, or for visual indication of alarms. A small Liquid Crystal Display (LCD) may also be used to display some important parameters.

5) Communication interfaces: The embedded systems may need to, interact with other embedded systems at they may have to transmit data to a desktop. To facilitate this, the embedded systems are provided with one or a few communication interfaces such as RS232, RS422, RS485, Universal Serial Bus (USB), IEEE 1394, Ethernet etc.

6) Application-specific circuitry: Sensors, transducers, special processing and control circuitry may be required fat an embedded system, depending on its application. This circuitry interacts with the processor to carry out the necessary work. The entire hardware has to be given power supply either through the 230 volts main supply or through a battery. The hardware has to design in such a way that the power consumption is minimized.

II. LITERATURE SURVEY

"Smart Waste Management Using IoT and GSM Technology" This study highlights the role of Internet of Things (IoT) and GSM technology in automating waste management processes. It discusses systems where sensors monitor dustbin fill levels and send notifications to designated authorities using GSM modules. The study emphasizes cost-effectiveness and reduced human intervention as key benefits. • "IoTBased Garbage Monitoring System for Smart Cities" Researchers proposed a solution to integrate smart sensors in dustbins to detect waste levels and communicate the data to municipal servers through IoT platforms. While primarily focused on Wi-Fi and Bluetooth-based communication, the paper outlines the challenges of network dependency in large-scale implementations, making GSM a more reliable alternative for remote areas. • "Smart Dustbin System for Efficient Waste Management in Urban Areas" This research introduces the use of ultrasonic sensors to measure waste levels in bins. The integration of GSM technology is suggested as an effective method for sending SMS alerts when bins are full. The study highlights the environmental benefits of optimized waste collection and reduced fuel consumption for waste collection vehicles. • "Arduino-Based IoT Applications for Smart Cities" The paper discusses the versatility of Arduino in developing smart city solutions, including waste management. It explores how Arduino can be interfaced with various sensors and communication modules, such as GSM, to create efficient systems. Case studies show the successful implementation of such systems in urban areas. • "A Comprehensive Review of

Smart Waste Management Systems" This literature reviews various smart waste management solutions using technologies like RFID, Zigbee, and GSM.

The review identifies GSM as a robust communication method for transmitting real-time data over long distances, making it suitable for applications in smart cities. • "Impact of Smart Dustbins on Urban Waste Management" This research highlights the potential impact of smart dustbins on urban cleanliness and hygiene. It evaluates systems where real-time waste level monitoring and SMS alerts reduce manual inspections and improve waste collection efficiency. The study also discusses how such systems align with sustainable development goals.

III. PROPOSED METHOD

A. *Overview*

The proposed system aims to develop a smart, self-sustaining sanitation management solution that ensures cleanliness, hygiene, and operational efficiency in public or community toilets. The core concept revolves around Internet of Things (IoT) and GSM-based communication, integrated with renewable solar power for continuous functionality, even in low-resource or off-grid environments.

By using sensors and automation, the system monitors sanitation parameters such as odor levels, waste accumulation, water tank levels, and human presence. In case of any abnormal condition—like gas buildup or an overflowing tank—the system automatically initiates actions such as flushing, activating a buzzer, or sending alerts via GSM or IoT cloud.

The goal of this method is not only to digitize sanitation monitoring but also to make it energy-efficient, low-maintenance, and user-friendly, ensuring better hygiene and sustainability in public facilities.

B. *Functional Objectives*

The proposed model is designed to:

1) Monitor sanitation parameters such as gas levels, water levels, and human activity in real time.
2) Provide instant alerts through SMS (GSM) and cloud notifications when thresholds are exceeded.
3) Automate cleaning actions like water flushing or deodorization to maintain hygiene.
4) Enable data visualization and analysis through cloud platforms like ThingSpeak.
5) Operate sustainably using solar power and battery backup.
6) Reduce manual inspection and enable predictive maintenance.

C. *System Architecture*

The architecture of the system follows a layered structure consisting of sensing, processing, communication, and response layers:

1) Sensing Layer – Comprises IR, Ultrasonic, and MQ135 gas sensors that collect data about the sanitation environment.
2) Processing Layer – The ESP32 microcontroller acts as the main processing unit, converting sensor data into readable values and executing decisions based on pre-defined logic.
3) Communication Layer – The system transmits data using Wi-Fi (to ThingSpeak) or GSM (via SIM800C module) for remote monitoring and SMS alerts.
4) Response Layer – Includes components like the DC pump, LCD display, and buzzer, which provide physical feedback and notifications to users and maintenance staff.

This modular architecture ensures scalability, allowing additional sensors or control features to be integrated easily.

D. *Working Principle*

1) Data Acquisition

The system begins by collecting real-time input from sensors.

- IR sensor detects human presence or activity.
- The Ultrasonic sensor measures the water or waste level in the tank.
- The MQ135 gas sensor monitors the air quality and detects the presence of harmful gases such as ammonia or methane.

2) Signal Processing: The ESP32 receives analog and digital signals from the sensors. These inputs are processed and compared with predefined threshold values.

3) Decision Making:

- If the gas concentration crosses the safety limit, the system activates the buzzer and sends an alert through the GSM module.
- If the tank reaches the maximum level, the ESP32 triggers the DC pump to drain water automatically.
- The LCD continuously updates with the latest readings and system status.

4) Data Transmission: The processed sensor data is uploaded to the ThingSpeak cloud using Wi-Fi for visualization and historical trend analysis. In areas without internet connectivity, the GSM module sends SMS alerts directly to the maintenance team.

5) Power Management: The entire system is powered by a solar panel, which charges a battery for energy storage. This ensures uninterrupted operation during cloudy days or nighttime.

E. Data Flow Description

The overall data flow of the system proceeds as follows:

1) Input Stage: Sensors (IR, MQ135, Ultrasonic) sense environmental parameters.
2) Processing Stage: ESP32 collects and processes the inputs.
3) Decision Stage: Logic conditions are applied (e.g., if gas > threshold → trigger buzzer).
4) Output Stage: Results are displayed on the LCD and communicated through GSM and IoT.
5) Storage and Visualization: Data is uploaded to ThingSpeak for cloud monitoring.
6) Action and Feedback: If critical conditions are detected, automatic cleaning or alert actions are executed.

This structured data flow enables continuous real-time monitoring and intelligent decision-making without human interference.

F. Algorithmic Approach

The implementation follows the following algorithmic flow:

1) Initialize all sensors and communication modules (ESP32 setup).
2) Read data from MQ135, IR, and Ultrasonic sensors.
3) Compare each sensor's output with defined threshold limits.
4) If condition met:
    - Activate buzzer for local alert.
    - Trigger DC pump if water level is high.
    - Send SMS alert via GSM.
    - Upload data to ThingSpeak for visualization.
5) Else: Continue normal monitoring and display values on the LCD.
6) Repeat the cycle continuously for real-time updates.

This algorithm ensures automated, fault-tolerant, and continuous operation of the sanitation monitoring system.

IV. SOFTWARE REQUIREMENTS

A. Arduino IDE

The Arduino Integrated Development Environment (IDE) serves as the main platform for writing, compiling, and uploading programs to the ESP32 microcontroller. It provides a user-friendly interface and supports languages such as C and C++. The IDE simplifies the integration of multiple sensors and modules by offering built-in libraries. It also allows serial monitoring to verify the communication and real-time response of the system. This makes it suitable for testing and deploying IoT-based embedded systems efficiently.

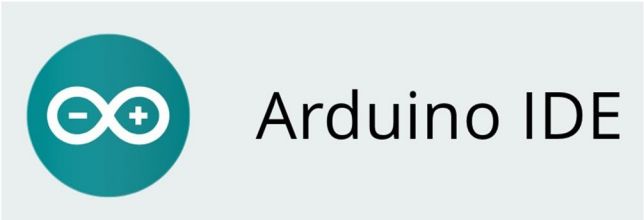


Fig 1

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue XI Nov 2025- Available at www.ijraset.com

B. Embedded C Language

Arduino boards are programmed in "C." C is a popular system programming language that has minimal execution time on hardware in comparison to other high-level programming languages. It's the reason most of the operating systems and several programming languages are built on C. Much like other microcontrollers, the AVR microcontrollers housed in Arduino boards are programmed in a subset of C. A general term for such subsets is "Embedded C" because they apply to programming embedded controllers. The language in which Arduino is programmed is a subset of C and it includes only those features of standard C that are supported by the Arduino IDE.

Fig 2

V. HARDWARE REQUIREMENTS

A. ESP32 Dev Board

The ESP32 microcontroller is the brain of the system. It is responsible for processing sensor inputs, controlling output devices, and managing communication with the cloud via Wi-Fi and GSM. With its dual-core processor and built-in wireless modules, it ensures efficient multitasking and reliable performance. The ESP32 coordinates data from sensors like the MQ135, PIR, and ultrasonic sensor and executes necessary actions such as activating pumps or sending alerts.

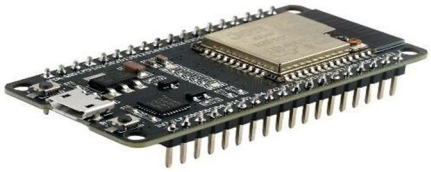


Fig 3

B. IR Sensor

The Infrared (IR) sensor is used for human detection inside the sanitation unit. When a person enters the toilet, the IR sensor detects motion and sends a signal to the ESP32. The controller then triggers an automatic flush after use. Its quick response and accuracy make it suitable for improving hygiene by automating the cleaning cycle.

Fig 4

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue XI Nov 2025- Available at www.ijraset.com

C. Ultrasonic Sensor

The ultrasonic sensor monitors the water or waste tank levels by measuring distance using sound waves. It helps determine whether the tank is full, empty, or at a critical level. The data from this sensor assists the ESP32 in activating the DC pump or sending alerts to prevent overflow or shortage. Its precision and non-contact operation make it ideal for sanitation monitoring.

Fig 5

D. MQ135 Gas Sensor

The MQ135 gas sensor is used to detect the presence of harmful or unpleasant gases such as ammonia, methane, and carbon dioxide. When gas levels exceed a defined threshold, the sensor sends an analog signal to the ESP32. The system then activates an air freshener or buzzer and sends an alert to the cloud via ThingSpeak or GSM. This enhances the air quality and safety of the sanitation unit.

Fig 6

E. LCD Display

A 16x2 Liquid Crystal Display (LCD) is used to show real-time system data such as gas concentration, tank levels, and network status. It provides immediate visual feedback to maintenance staff or users without requiring internet access. The LCD is directly connected to the ESP32 and updated continuously based on sensor readings.

Fig 7

F. Solar Panel

The solar panel powers the entire system, ensuring it remains functional even in areas without reliable electricity. It charges the battery, which in turn supplies regulated power to the ESP32, sensors, and GSM module. Using renewable energy makes the system sustainable, low-maintenance, and environmentally friendly.

Fig 8

G. DC Pump

The DC pump is responsible for flushing or spraying water in the sanitation system. It is controlled through a relay that receives signals from the ESP32. When the ultrasonic sensor detects low water levels or the PIR sensor indicates post-use, the ESP32 activates the pump automatically. This reduces manual intervention and ensures consistent sanitation.

Fig 9

H. GSM SIM800C Module

The GSM SIM800C module provides cellular connectivity for sending SMS alerts and data transmission in the absence of Wi-Fi. It allows the system to send notifications about abnormal conditions, such as gas leaks or tank overflows, directly to maintenance personnel. Its inclusion enhances system reliability and remote accessibility.

Fig 10

I. Buzzer

The buzzer functions as an alarm system to alert users or maintenance staff in case of abnormal readings, such as gas detection or high tank levels. It produces an audible signal that draws immediate attention, ensuring timely corrective actions.

Fig 11

J. Relay Module

A relay is an electrical switch that can be used to control devices and systems that use higher voltages. In the case of module relay, the mechanism is typically an electromagnet. The relay board input voltage is usually DC. However, the electrical load that a relay will control can be either AC, but essentially within the limit levels that the relay is designed for. A relay module is available in an array of input voltage ratings: It can be a 3.2V or 5V relay module for low power switching, or it can be a 12 or 24V relay module for heavy-duty systems. The relay module infomation is normally printed on the surface of the device for ready reference. This include the input voltage rating, switch voltage, and current limit.

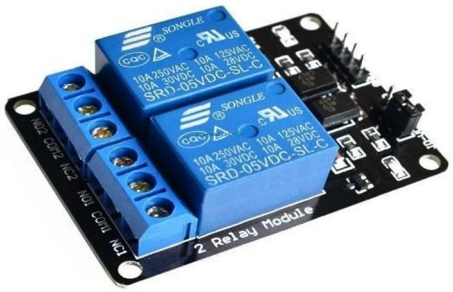


Fig 12

VI. PROJECT IMPLEMENTATION

A. Hardware Implementation

The hardware was designed and assembled on a prototype board for testing. The ESP32 acts as the central controller, connected to the MQ135 gas sensor, Ultrasonic sensor, IR sensor, LCD display, buzzer, GSM module, and DC pump.

1) The ESP32 GPIO pins interface with each sensor for data collection.
2) The LCD displays real-time readings of gas concentration, tank levels, and user activity.
3) The GSM module uses UART communication to send SMS alerts.
4) The solar panel supplies power, while a voltage regulator maintains 5V and 3.3V for stable operation.
5) The buzzer provides audible alerts during abnormal conditions.

The hardware is enclosed in a protective casing to prevent environmental damage, making it suitable for field installation.

B. Software Implementation

The software is developed using the Arduino IDE. The programming logic is written in C/C++, and several libraries are imported to simplify tasks such as sensor reading and data transmission.

Key libraries used include:

1) WiFi.h – Enables IoT connectivity
2) HTTPClient.h – Sends data to cloud
3) LiquidCrystal.h – LCD display control
4) SoftwareSerial.h – GSM communication
5) NewPing.h – Ultrasonic distance sensing
6) Wire.h – I²C data transfer
7) MQ135.h – Gas and air quality sensing

The program continuously reads data from sensors, processes it, and performs actions based on predefined conditions. It then updates the LCD and uploads data to the cloud for monitoring.

C. Testing and Validation

The system was tested under multiple real-world conditions to ensure accuracy and stability.

1) Sensor Calibration: Verified the accuracy of gas and ultrasonic sensors.
2) Power Test: Confirmed stable operation under solar and battery modes.
3) Communication Test: Ensured successful IoT uploads and SMS delivery.

4) Alert Test: Checked buzzer and notification triggers under threshold breach.
5) Reliability Test: Continuous 24-hour operation confirmed system consistency.
The system performed reliably across all tests, proving its suitability for practical deployment.

D. Workflow Summary
1) The system initializes all modules after power-up.
2) Sensors continuously send data to the ESP32.
3) The microcontroller analyzes data and triggers responses.
4) Alerts are sent via GSM or displayed on the cloud dashboard.
5) The solar-powered system continues to function autonomously with minimal maintenance.

E. Outcome
The implemented model successfully demonstrated real-time sanitation monitoring, automated alerts, and remote data visualization. It proved to be cost-efficient, sustainable, and scalable, making it suitable for large-scale adoption in urban and rural sanitation systems.

VII. FLOW CHART & BLOCK DIAGRAM

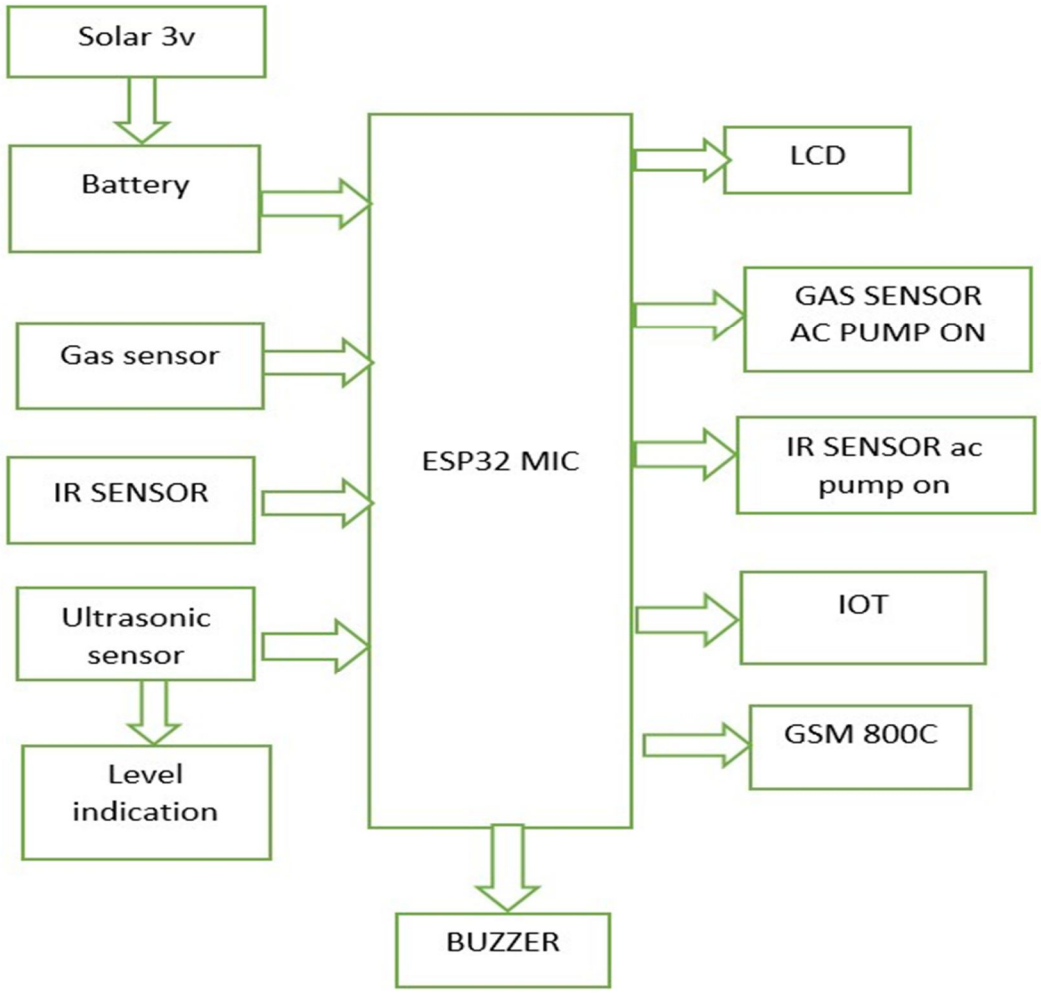


Fig 13

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue XI Nov 2025- Available at www.ijraset.com

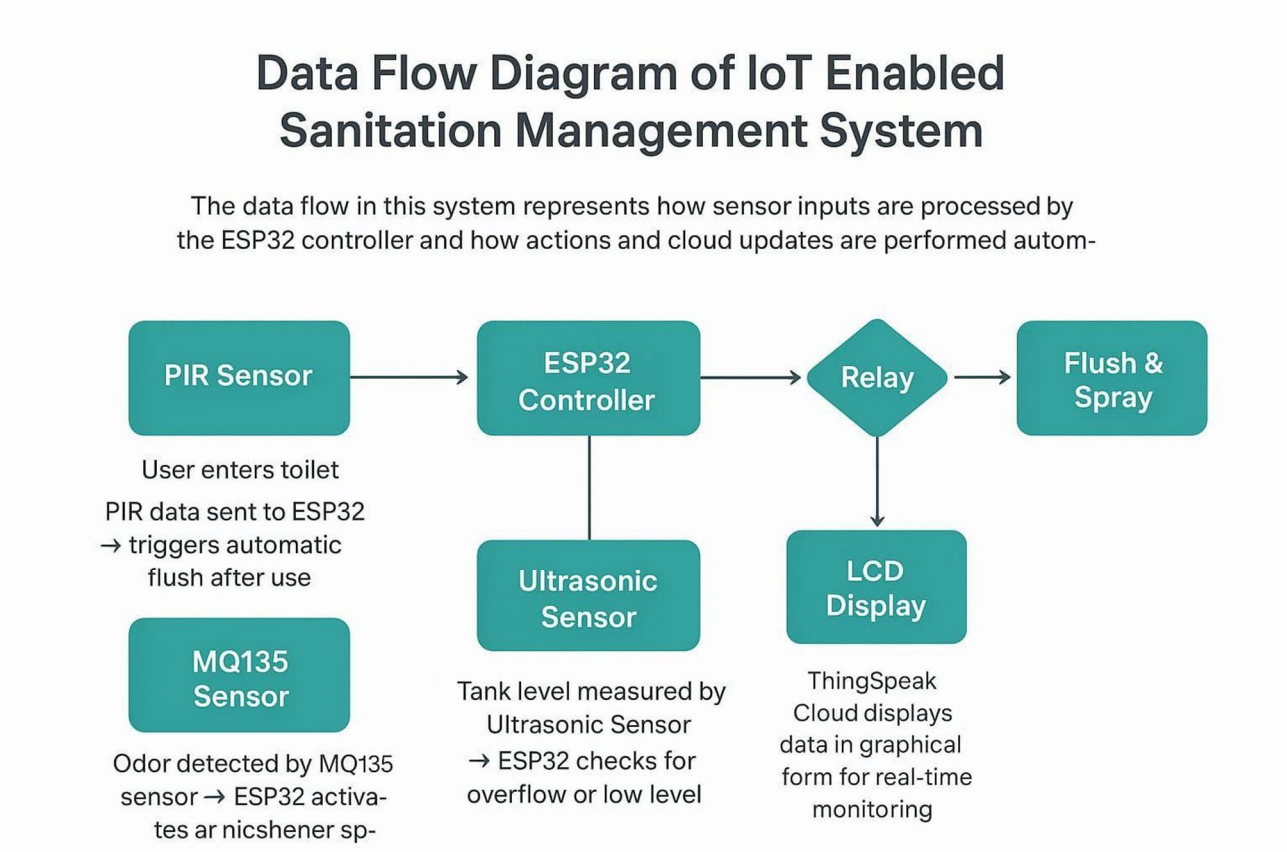


Fig 14

VIII. APPENDEX

```
#include <LiquidCrystal.h>
#include <stdio.h>

//LiquidCrystal lcd(6, 7, 5, 4, 3, 2);  //rs,en,d4,d5,d6,d7
LiquidCrystal lcd(13, 12, 14, 27, 26, 25);

#include <Wire.h>
 #include <WiFi.h>
#include <HTTPClient.h>

HTTPClient http;
 const char *ssid = "iotserver"; const char *password = "iotserver123";

int httpResponseCode;
String servername = "http://projectsfactoryserver.in/storedata.php?name=";
String accountname = "iot1529";
String field1 = "&s1=";
String field2 = "&s2=";
String field3 = "&s3=";
String field4 = "&s4=";
String field5 = "&s5=";
String payload="";
```



```
int lvlv=0;
String person_string="";
String odour_string="";
String waterflush_string="";
String odourpump_string="";


int sti=0;
String inputString = "";         // a string to hold incoming data boolean stringComplete = false;  // whether the string is complete char
gchr,rcv,pastnumber[11]; int tempc=0,humc=0;

int buzzer = 23; int gas    = 19;

const int trigPin = 5; const int echoPin = 18;

int ir    = 21;

int pump1 = 17; int pump2 = 16;


//#define RXD2 16
//#define TXD2 17


int rtr1=0; int dist1=0,dist2=0,dist3,sts1=0,sts2=0; long duration; int distanceCm, distanceInch; int distancemm=0;

unsigned int ultra_dist()
{int ud=0;

  digitalWrite(trigPin, LOW);    delayMicroseconds(2);    digitalWrite(trigPin, HIGH);    delayMicroseconds(10);
digitalWrite(trigPin, LOW);    duration = pulseIn(echoPin, HIGH);
  //distanceCm= duration*0.034/2;
  //ud = distanceCm;

  distancemm = (duration*0.17);    ud = distancemm;

  return ud;
}


char gpsval[50];
// char dataread[100] = "";
// char lt[15],ln[15];


int i=0,k=0,lop=0; int  gps_status=0; float latitude=0;
float logitude=0;
String Speed="";
String gpsString=""; char *test="$GPRMC";
```

```
//int hbtc=0,hbtc1=0,rtrl=0;

unsigned char gv=0,msg1[10],msg2[11];  float lati=0,longi=0; unsigned int lati1=0,longi1=0; unsigned char flat[5],flong[5]; char finallat[10]="17.5602\0",finallong[10]="078.4495\0";
//17.5602° N, 78.4495° E
String finallat1="";
String finallong1="";

int ii=0,rchkr=0;

void iot_send()
{       lcd.setCursor(15,1);lcd.print("U");
     http.begin(servername + accountname + field1 + String(lvlv) + field2 + person_string + field3 + odour_string + field4 + waterflush_string + field5 + odourpump_string);
     httpResponseCode = http.GET();       if(httpResponseCode>0)
       {
        payload="";
        //Serial.print("HTTP Response code: ");         //Serial.println(httpResponseCode);         payload = http.getString();
        //Serial.println(payload);
       }       else         {
        ;
        //Serial.print("Error code: ");
        //Serial.println(httpResponseCode);
       }
    delay(3000);
      //lcd.setCursor(0,1);lcd.print("Emergency  ");      lcd.setCursor(15,1);lcd.print(" ");
}
void beep()
{   digitalWrite(buzzer, LOW);delay(2000); digitalWrite(buzzer, HIGH);delay(200);
} void okcheck0()
{   unsigned char rcr;   do{       rcr = Serial.read();
    }while(rcr != 'K');
} void gsm_send(String strs)
{
    lcd.setCursor(15,1);lcd.print("G");
    delay(4000);  delay(4000); delay(4000);  delay(4000);
    Serial.write("AT+CMGS=\"");
    Serial.write(pastnumber);
    Serial.write("\"\r\n"); delay(3000);
    Serial.print(strs);                Serial.write(0x1A);                          delay(4000);       delay(4000);       delay(4000);    delay(4000);
lcd.setCursor(15,1);lcd.print(" "); }
void setup()
{
  Serial.begin(9600);
  //Serial2.begin(9600, SERIAL_8N1, RXD2, TXD2);

  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output   pinMode(echoPin, INPUT); // Sets the echoPin as an Input
```

```
 pinMode(pump1, OUTPUT);pinMode(pump2, OUTPUT);   pinMode(buzzer, OUTPUT);
 pinMode(gas, INPUT);pinMode(ir, INPUT);

 digitalWrite(pump1, LOW);digitalWrite(pump2, LOW);   digitalWrite(buzzer, HIGH);


 lcd.begin(16, 2);lcd.cursor();   lcd.print("   Welcome   ");   //lcd.setCursor(0,1);   //lcd.print("   Management ");   delay(2500);

 WiFi.begin(ssid, password);
 //Serial.println("Connecting");   while(WiFi.status() != WL_CONNECTED)
    {
      delay(500);
    }
 //Serial.println(WiFi.localIP());   delay(3000);

 gsminit();

 lcd.clear();   lcd.print("L:"); //2,0   lcd.setCursor(8,0);   lcd.print("O:"); //10,0

 lcd.setCursor(0,1);   lcd.print("P:"); //2,1  // serialEvent();
}

int cntlmk=0; int dista=0;

void loop()  {   dista=0;   dist1=0;   dist2=0;

 for(rtr1=0;rtr1<5;rtr1++)
   {     dista = ultra_dist();     dist1 = (dist1 + dista);       delay(10);
   }  dist1 = (dist1/5);  if(dist1 > 220)
  {    dist2 = 220;    }  else    {     dist2 = dist1;
  }

 lvlv = map(dist2,220,30,0,100);  if(lvlv > 100){lvlv=100;}
 lcd.setCursor(2,0);convertl(lvlv);lcd.print("%");

 if(lvlv < 10)    {sts1++;       if(sts1 >= 2){sts1=2;}     if(sts1 == 1)       {beep();        iot_send();        String gsm_string="";
gsm_string = "Level_Aboutto_Empty_" + String(lvlv);         gsm_send(gsm_string);
    }    }  else    {     sts1=0;
  }

 person_string="";  person_string="---";  waterflush_string="";  waterflush_string="---";  if(digitalRead(ir) == LOW)
{lcd.setCursor(2,1);lcd.print("Entry");     delay(500);     while(digitalRead(ir) == LOW);       person_string="";       person_string =
"Entered";       waterflush_string = "";       waterflush_string = "Water_Flush_ON";       digitalWrite(pump1, HIGH);       iot_send();
delay(5000);
   digitalWrite(pump1, LOW);

    String gsm_string="";
    gsm_string = "Entry_Water_Flush_ON";       gsm_send(gsm_string);

   lcd.setCursor(2,1);lcd.print("-----");
```

```
  }

 odour_string="";  odourpump_string="";  if(digitalRead(gas) == LOW)
  {    lcd.setCursor(10,0);lcd.print("Det");     odour_string = "Detected";    odourpump_string = "Activated";    beep();
digitalWrite(pump2, HIGH);    iot_send();      delay(5000);    digitalWrite(pump2, LOW);

     String gsm_string="";      gsm_string = "Odour_Det_Odour_Pump_Activated";       gsm_send(gsm_string);
  }
 if(digitalRead(gas) == HIGH)
  {    lcd.setCursor(10,0);lcd.print("---");     odour_string = "---";    odourpump_string = "---";
  }

}

/*
void serialEvent()
{    while(Serial.available())
       {
        char inChar = (char)Serial.read();
          //sti++;
          //inputString += inChar;          if(inChar == '*')          {sti=1;
            inputString += inChar;            //  stringComplete = true;
          // gchr = inputString[sti-1]
          }          if(sti == 1)
          {
            inputString += inChar;
          }
        if(inChar == '#')          {sti=0;            stringComplete = true;                }
       }
} */ int readSerial(char result[])
{   int i = 0;   while (1)    {     while (Serial.available() > 0)
   {      char inChar = Serial.read();      if (inChar == '\n')
       {          result[i] = '\0';          Serial.flush();          return 0;         }
     if (inChar != '\r')
       {          result[i] = inChar;          i++;
       }
   }
  }
}


void gsminit()
{
  Serial.write("AT\r\n");                 okcheck0();
  Serial.write("ATE0\r\n");               okcheck0();
  Serial.write("AT+CMGF=1\r\n");          okcheck0();
  Serial.write("AT+CNMI=1,2,0,0\r\n");    okcheck0();
  Serial.write("AT+CSMP=17,167,0,0\r\n");  okcheck0();

  lcd.clear();
```

```
 lcd.print("SEND MSG STORE");  lcd.setCursor(0,1);  lcd.print("MOBILE NUMBER");  do{  rcv = Serial.read();
}while(rcv != '*');  readSerial(pastnumber);

 lcd.clear();  lcd.print(pastnumber); pastnumber[10]='\0';

  delay(4000);  delay(4000);  Serial.write("AT+CMGS=\"");
  Serial.write(pastnumber);
  Serial.write("\"\r\n"); delay(3000);
  Serial.write("Mobile no. registered\r\n");
  Serial.write(0x1A);  delay(4000);  delay(4000);
}

void convertl(unsigned int value)
{  unsigned int a,b,c,d,e,f,g,h;

    a=value/10000;  b=value%10000;  c=b/1000;  d=b%1000;  e=d/100;  f=d%100;  g=f/10;  h=f%10;


    a=a|0x30;  c=c|0x30;  e=e|0x30;  g=g|0x30;  h=h|0x30;


 // lcd.write(a);
 // lcd.write(c);
  lcd.write(e);  lcd.write(g);  lcd.write(h);
} void convertk(unsigned int value)
{  unsigned int a,b,c,d,e,f,g,h;

    a=value/10000;  b=value%10000;  c=b/1000;  d=b%1000;  e=d/100;  f=d%100;  g=f/10;  h=f%10;

     a=a|0x30;  c=c|0x30;  e=e|0x30;  g=g|0x30;  h=h|0x30;


 // lcd.write(a);
 // lcd.write(c);
 // lcd.write(e);  // lcd.write(g);  lcd.write(h);
}


void gpsEvent()
{  gpsString="";  while(1)
  {
  //while (gps.available()>0)  //Serial incoming data from GPS  while (Serial.available() > 0)
  {
   //char inChar = (char)gps.read();  char inChar = (char)Serial.read();  gpsString+= inChar;  //store incoming data
from GPS to temparary string str[]  i++;
  // Serial.print(inChar);
   if (i < 7)
   {  if(gpsString[i-1] != test[i-1])  //check for right string
    {  i=0;
     gpsString="";
```

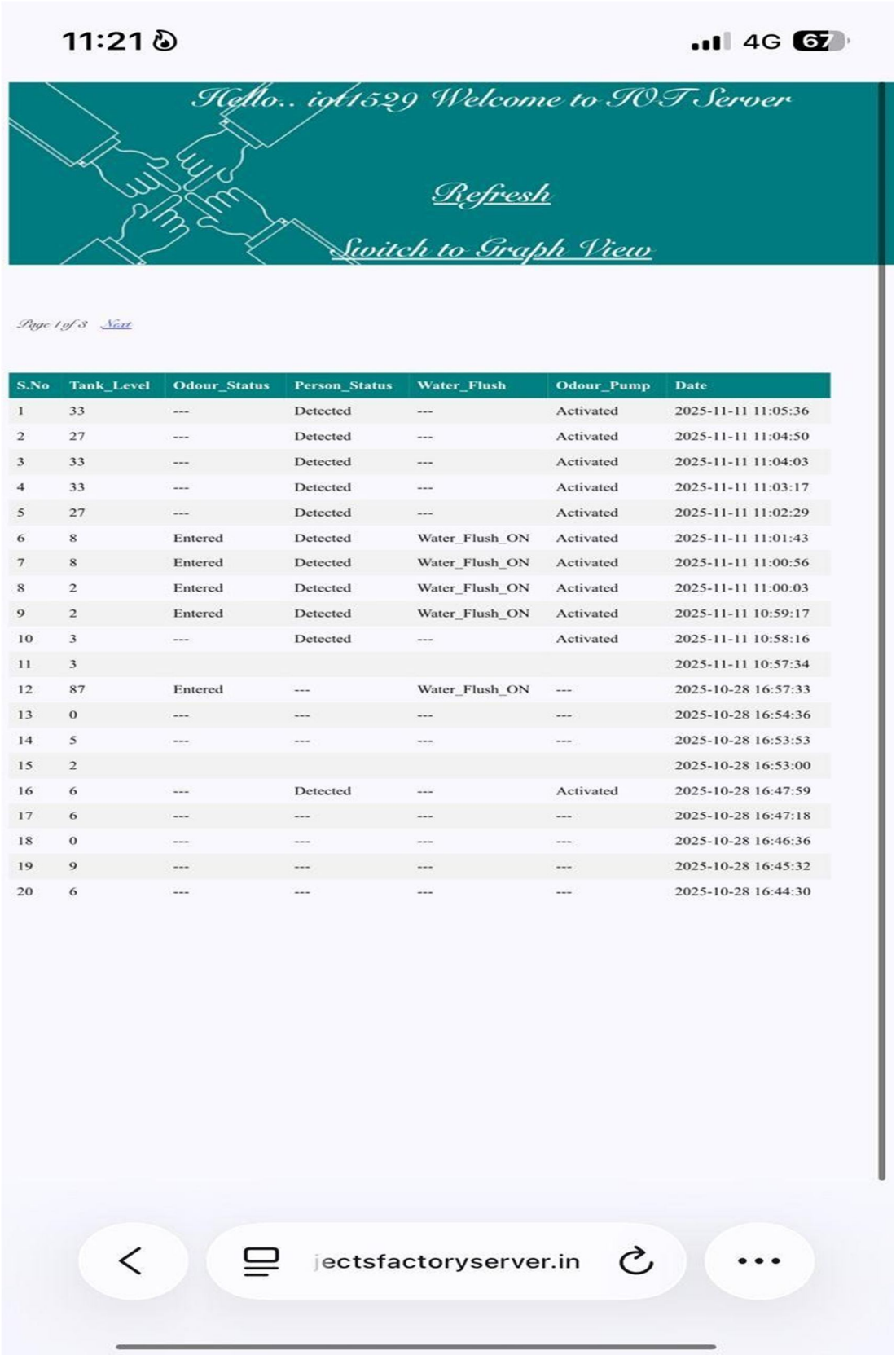
```
      }
     }     if(inChar=='\r')
    {      if(i>60)
     {
       gps_status=1;        break;      }      else      {        i=0;
     }
    }   }
   if(gps_status)     break;
  }
}

void get_gps()
{

 lcd.clear();
 lcd.print("Getting GPS Data");   lcd.setCursor(0,1);   lcd.print("Please Wait.....");

  gps_status=0;    int x=0;    while(gps_status==0)
  {     gpsEvent();     int str_lenth=i;     coordinate2dec();     i=0;x=0;     str_lenth=0;
  }
}

void coordinate2dec()
{
 String lat_degree="";     for(i=17;i<=18;i++)              lat_degree+=gpsString[i];

 String lat_minut="";      for(i=18;i<=19;i++)              lat_minut+=gpsString[i];      for(i=21;i<=22;i++)
lat_minut+=gpsString[i];



 String log_degree="";     for(i=29;i<=31;i++)       log_degree+=gpsString[i];   String log_minut="";     for(i=32;i<=33;i++)
log_minut+=gpsString[i];     for(i=35;i<=36;i++)       log_minut+=gpsString[i];


   Speed="";     for(i=42;i<45;i++)         //extract longitude from string       Speed+=gpsString[i];

    float minut= lat_minut.toFloat();      minut=minut/60;      float degree=lat_degree.toFloat();      latitude=degree+minut;

    minut= log_minut.toFloat();      minut=minut/60;      degree=log_degree.toFloat();      logitude=degree+minut;
}

/*
void coordinate2dec()
{
 String lat_degree="";     for(i=19;i<=20;i++)              lat_degree+=gpsString[i];

 String lat_minut="";      for(i=21;i<=22;i++)              lat_minut+=gpsString[i];      for(i=24;i<=25;i++)
lat_minut+=gpsString[i];
```

```
 String log_degree="";    for(i=32;i<=34;i++)      log_degree+=gpsString[i];   String log_minut="";    for(i=35;i<=36;i++)
log_minut+=gpsString[i];    for(i=38;i<=39;i++)      log_minut+=gpsString[i];


   Speed="";    for(i=45;i<48;i++)        //extract longitude from string      Speed+=gpsString[i];

    float minut= lat_minut.toFloat();     minut=minut/60;     float degree=lat_degree.toFloat();     latitude=degree+minut;

    minut= log_minut.toFloat();     minut=minut/60;     degree=log_degree.toFloat();     logitude=degree+minut;
} */ void gps_convert()
{   if(gps_status)
  {
 // Serial.println(gpsString);

 if(gpsString[0] == '$' && gpsString[1] == 'G' && gpsString[2] == 'P' && gpsString[3] == 'R' && gpsString[4] == 'M' &&
gpsString[5] == 'C')
   {
    //
Serial.println("Don1111111111111111111111111111111111111111111111111111\r\n")
;
   //
Serial.write(gpsString[18]);Serial.write(gpsString[19]);Serial.write(gpsString[20]);Se
rial.write(gpsString[21]);Serial.write(gpsString[22]);
    //lcd.setCursor(0,0);     for(ii=0;ii<9;ii++)
      {
       //lcd.write(gpsString[19+ii]);            msg1[ii] = gpsString[19+ii];
       //Serial.write(msg1[ii]);
      }
      //Serial.println("\r\n");     //lcd.setCursor(0,1);      for(ii=0;ii<10;ii++)
      {
       //lcd.write(gpsString[32+ii]);
       msg2[ii] = gpsString[32+ii];
      // Serial.write(msg2[ii]);
      }

// Serial.println(msg1);
// Serial.println(msg2);

      //lati = (((msg1[2]-48)*100000) +((msg1[3]-48)*10000) + ((msg1[5]-48)*1000)
+ ((msg1[6]-48)*100) + ((msg1[7]-48)*10) + (msg1[8]-48));
      //longi = (((msg2[3]-48)*100000) + ((msg2[4]-48)*10000) + ((msg2[6]48)*1000) + ((msg2[7]-48)*100) + ((msg2[8]-48)*10) +
(msg2[9]-48));

      lati = (((msg1[2]-48)*1000) + ((msg1[3]-48)*100) + ((msg1[5]-48)*10) +
(msg1[6]-48));       longi = (((msg2[3]-48)*1000) + ((msg2[4]-48)*100) + ((msg2[6]-48)*10) + (msg2[7]-48));

    // converts(lati);Serial.write("-");
    // converts(longi);Serial.write("\r\n");
```

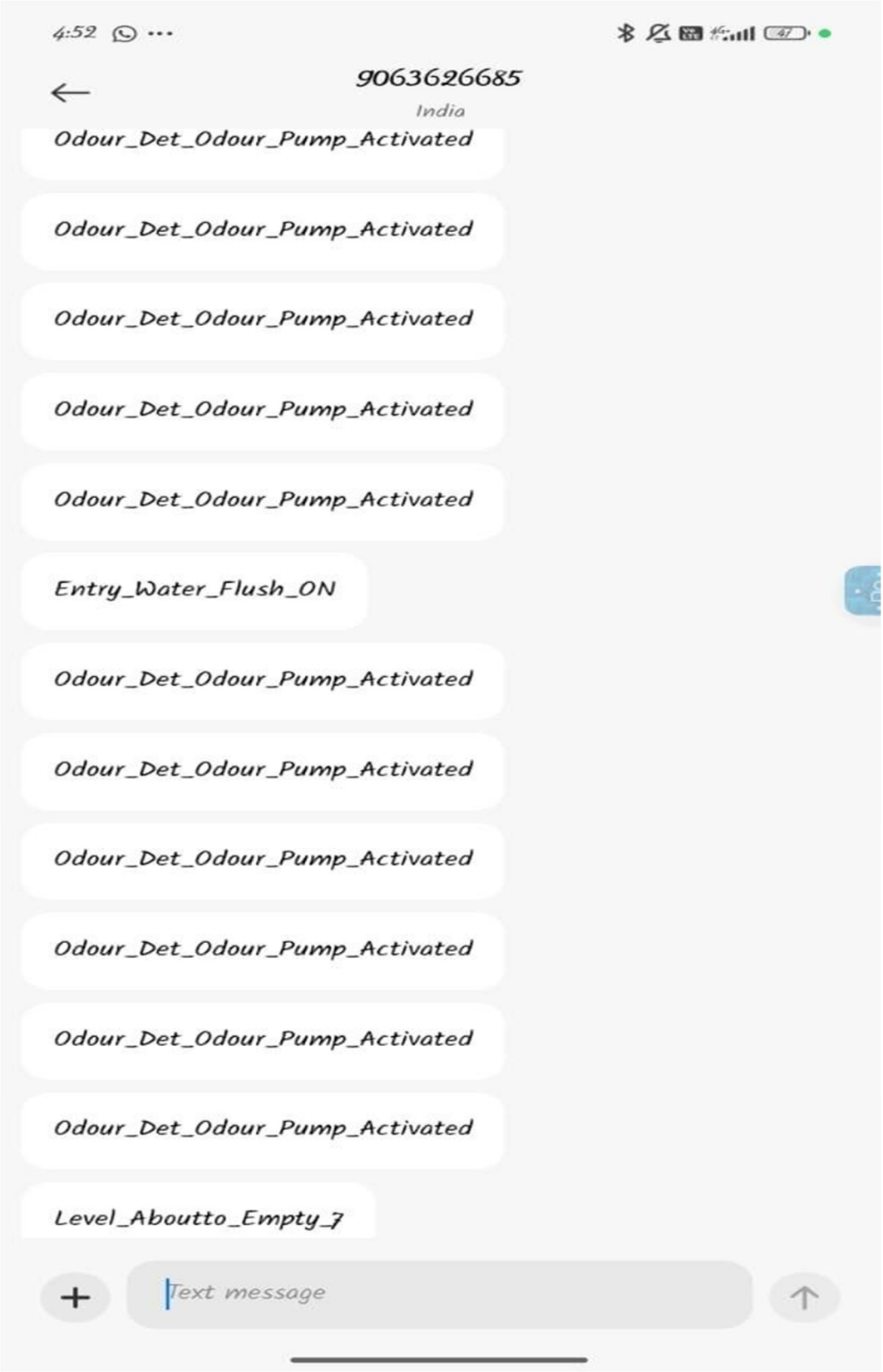
```
      lati = (lati/60);  longi = (longi/60);

      lati = (lati*100); longi = (longi*100);       lati1 = lati;      longi1 = longi;

// Serial.write("After ");
  //      converts(lati1);Serial.write("-");
    //   converts(longi1);Serial.write("\r\n");


            convlat(lati); convlong(longi);       finallat[0] = msg1[0];       finallat[1] = msg1[1];       finallat[2] = '.';       finallat[3] =
flat[0]; finallat[4] = flat[1];finallat[5] = flat[2];finallat[6] = flat[3];finallat[7] = '\0';


       finallong[0] = msg2[0];       finallong[1] = msg2[1];       finallong[2] = msg2[2];       finallong[3] = '.';       finallong[4] =
flong[0];finallong[5] = flong[1];finallong[6] = flong[2];finallong[7] = flong[3];finallong[8] = '\0';


    }
  }
}

 void convlat(unsigned int value)
     {             unsigned int a,b,c,d,e,f,g,h;

     a=value/10000;      b=value%10000;      c=b/1000;      d=b%1000;      e=d/100;      f=d%100;      g=f/10;      h=f%10;

      a=a|0x30;                  c=c|0x30;      e=e|0x30;      g=g|0x30;                  h=h|0x30;

  // dlcd(a);
//   dlcd(c);dlcd(e); dlcd(g);dlcd(h);//lcddata('A');//lcddata(' ');lcddata(' ');

                     flat[0] = c;              flat[1] = e;              flat[2] = g;              flat[3] = h;


         }

 void convlong(unsigned int value)
     {             unsigned int a,b,c,d,e,f,g,h;

     a=value/10000;      b=value%10000;      c=b/1000;      d=b%1000;      e=d/100;      f=d%100;      g=f/10;      h=f%10;

      a=a|0x30;                  c=c|0x30;      e=e|0x30;      g=g|0x30;                  h=h|0x30;

  // dlcd(a);
//   dlcd(c);dlcd(e); dlcd(g);dlcd(h);//lcddata('A');//lcddata(' ');lcddata(' ');
                         flong[0] = c;              flong[1] = e;              flong[2] = g;              flong[3] = h;

         }
```

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue XI Nov 2025- Available at www.ijraset.com

IX. OUTPUT

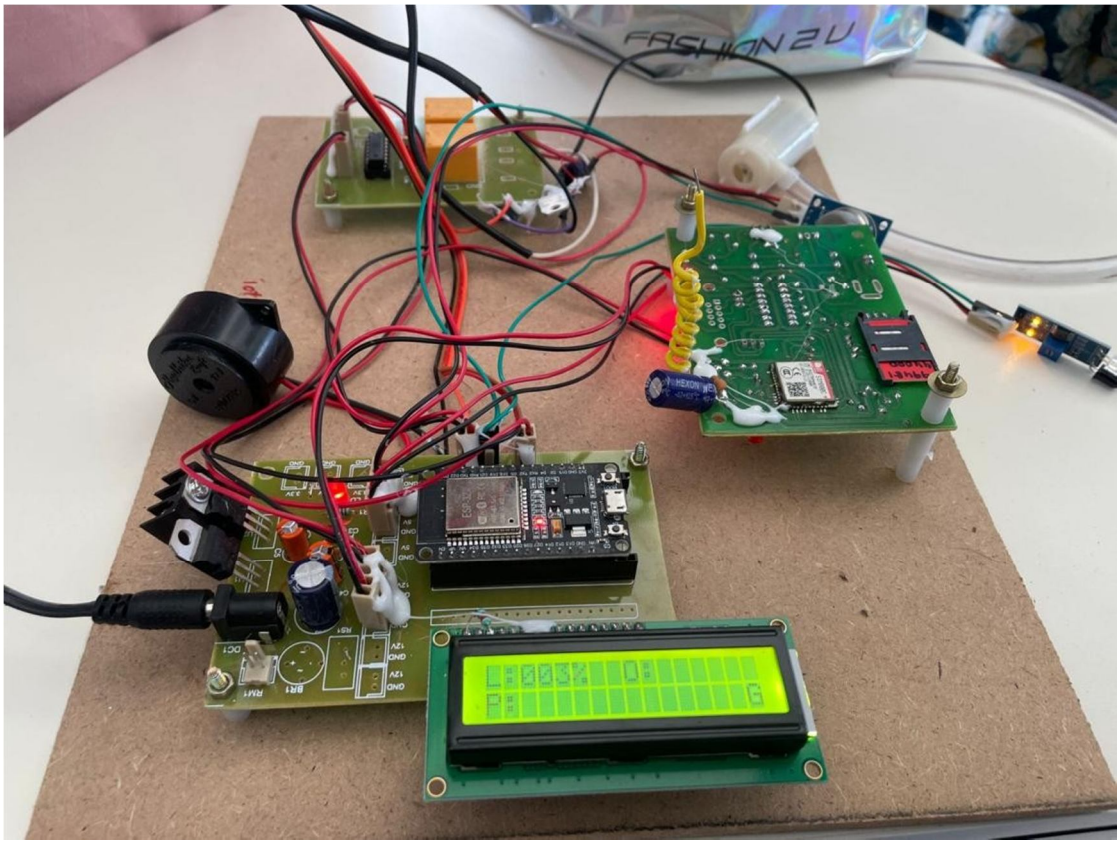


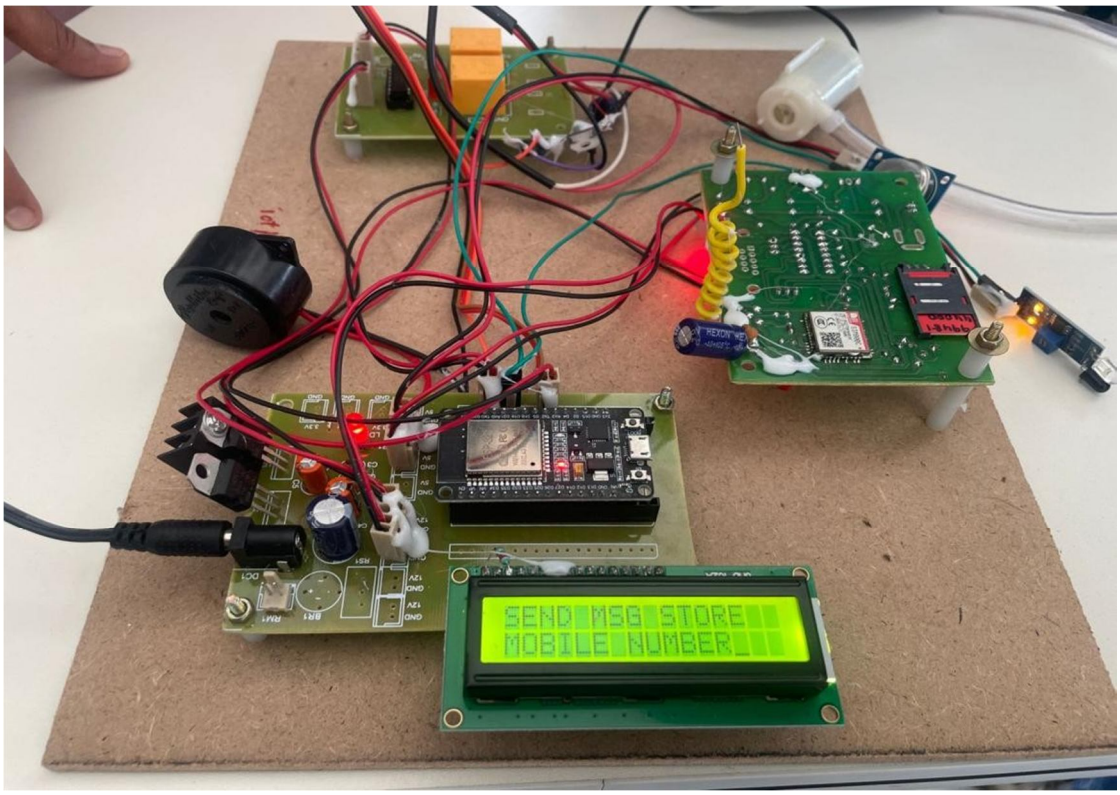
S.No	Tank_Level	Odour_Status	Person_Status	Water_Flush	Odour_Pump	Date
1	33	---	Detected	---	Activated	2025-11-11 11:05:36
2	27	---	Detected	---	Activated	2025-11-11 11:04:50
3	33	---	Detected	---	Activated	2025-11-11 11:04:03
4	33	---	Detected	---	Activated	2025-11-11 11:03:17
5	27	---	Detected	---	Activated	2025-11-11 11:02:29
6	8	Entered	Detected	Water_Flush_ON	Activated	2025-11-11 11:01:43
7	8	Entered	Detected	Water_Flush_ON	Activated	2025-11-11 11:00:56
8	2	Entered	Detected	Water_Flush_ON	Activated	2025-11-11 11:00:03
9	2	Entered	Detected	Water_Flush_ON	Activated	2025-11-11 10:59:17
10	3	---	Detected	---	Activated	2025-11-11 10:58:16
11	3					2025-11-11 10:57:34
12	87	Entered	---	Water_Flush_ON	---	2025-10-28 16:57:33
13	0	---	---	---	---	2025-10-28 16:54:36
14	5	---	---	---	---	2025-10-28 16:53:53
15	2					2025-10-28 16:53:00
16	6	---	Detected	---	Activated	2025-10-28 16:47:59
17	6	---	---	---	---	2025-10-28 16:47:18
18	0	---	---	---	---	2025-10-28 16:46:36
19	9	---	---	---	---	2025-10-28 16:45:32
20	6	---	---	---	---	2025-10-28 16:44:30

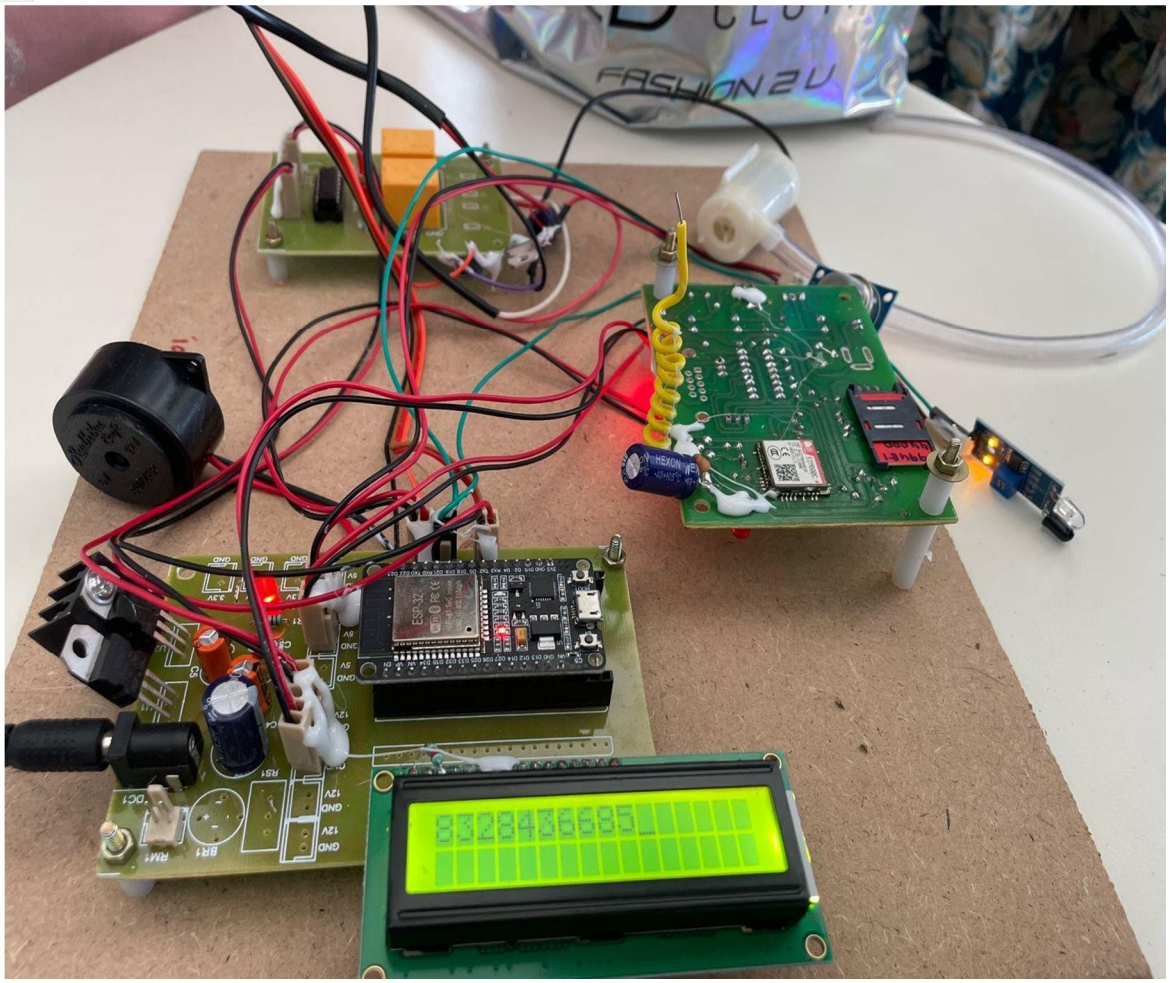

4:52
9063626685
India
Odour_Det_Odour_Pump_Activated
Odour_Det_Odour_Pump_Activated
Odour_Det_Odour_Pump_Activated
Odour_Det_Odour_Pump_Activated
Odour_Det_Odour_Pump_Activated
Entry_Water_Flush_ON
Odour_Det_Odour_Pump_Activated
Odour_Det_Odour_Pump_Activated
Odour_Det_Odour_Pump_Activated
Odour_Det_Odour_Pump_Activated
Odour_Det_Odour_Pump_Activated
Odour_Det_Odour_Pump_Activated
Level_Aboutto_Empty_7
Text message

Fig(s) 15

X. HARDWARE RESULT







Fig(s) 16

XI. ADVANTAGES

1) Real-Time Monitoring: The system continuously monitors parameters such as air quality, human presence, and water tank levels in real time. Through IoT integration, this information is instantly updated on the ThingSpeak cloud platform, allowing administrators to make timely decisions without requiring on-site supervision. This reduces response time and enhances overall maintenance efficiency.
2) Automation and Reduced Human Intervention :By utilizing sensors such as PIR, MQ135, and Ultrasonic, the system automates crucial functions like flushing, odor control, and water management. This automation minimizes the dependency on manual labor and ensures consistent sanitation operations even during staff shortages or emergencies.
3) Energy Efficiency through Solar Power: The incorporation of a solar panel ensures that the system operates independently of traditional power sources. This makes it ideal for remote or off-grid areas where electricity is unreliable. Solar energy reduces operational costs and supports sustainability initiatives, aligning with smart city and green technology goals.
4) Enhanced Hygiene and Odor Control: The automatic activation of air fresheners or exhaust fans through the MQ135 sensor significantly improves air quality. The system ensures that foul odors are promptly neutralized, maintaining a pleasant and hygienic environment. Automated flushing and cleaning cycles prevent the accumulation of waste and bacteria.
5) Cost-Effective Maintenance: The system provides predictive and preventive maintenance alerts via GSM messages and IoT dashboards. Early detection of issues, such as tank overflow or gas buildup, helps reduce repair costs and prevent damage. Since data is stored and analyzed remotely, maintenance personnel can prioritize and manage resources more efficiently.
6) Data Analytics and Cloud Integration: All sensor data is stored on the cloud platform (ThingSpeak), allowing for historical data analysis and visualization. This helps identify trends in usage, resource consumption, and maintenance frequency. Data-driven insights can guide future improvements in sanitation infrastructure and policy formulation.

7) Scalability and Customization: The system is highly modular — additional sensors or functionalities (like humidity, temperature, or camera modules) can be easily integrated. This flexibility allows it to be customized for different sanitation setups, including public toilets, schools, hospitals, and community centers.
8) Reliability through Dual Communication Channels: Even if Wi-Fi connectivity fails, the GSM module ensures that SMS alerts are sent to concerned authorities. This dual communication mechanism enhances the reliability of the system and ensures that critical updates are never missed.
9) Eco-Friendly Operation: By integrating solar energy and automating water usage, the system contributes to environmental conservation. Reduced water wastage, efficient power utilization, and minimal manual intervention make it an eco-friendly sanitation management solution.
10) Improved Public Health and Safety: Clean and well-maintained sanitation facilities reduce the risk of diseases such as diarrhea, cholera, and typhoid, especially in densely populated or underprivileged areas. Continuous monitoring ensures that sanitation standards are maintained, directly improving public health and well-being.

XII. APPLICATIONS

1) Public Toilets in Urban and Rural Areas: The system can be deployed in public restrooms, bus stations, railway platforms, and marketplaces to ensure automated maintenance and cleanliness. Real-time alerts help municipal authorities manage sanitation more effectively and respond to issues promptly.
2) Slum and Low-Income Area Sanitation: In underdeveloped or slum regions where manual monitoring is difficult, the IoT-enabled system provides a low-cost and sustainable solution. Solar-powered operation ensures functionality even without grid electricity, promoting hygiene and reducing health risks.
3) Smart City Infrastructure: This project aligns with the Smart City Mission by contributing to the digital transformation of urban infrastructure. The integration of IoT, GSM, and cloud computing enhances civic sanitation management, enabling centralized control and analytics-driven governance.
4) Hospitals and Healthcare Facilities: Maintaining hygiene in hospitals is critical to preventing infections. The system can monitor air quality and sanitation status in washrooms and patient areas, automatically triggering cleaning cycles or air purification when necessary.
5) Educational Institutions: The project can be implemented in schools and colleges to ensure hygienic restroom conditions. Automated flushing and air purification not only improve cleanliness but also reduce water and energy wastage.
6) Corporate and Commercial Buildings: Smart sanitation systems can be integrated into corporate offices, malls, and industrial premises to maintain high hygiene standards with minimal manual effort. The system's data can also be used to schedule maintenance during non-working hours.
7) Temporary Event Venues: The system is suitable for large gatherings, such as festivals, fairs, and sports events, where temporary sanitation facilities are installed. IoT-based monitoring ensures efficient maintenance even during high usage periods.
8) Remote and Off-Grid Locations: Due to its solar-based power supply and GSM connectivity, the system is ideal for rural and remote installations where internet and power availability are limited. It provides an autonomous sanitation solution for remote schools, tourist spots, and community centers.
9) Government and NGO Initiatives: The system supports initiatives aimed at improving rural sanitation and public health, such as the Swachh Bharat Mission. NGOs and local authorities can use the collected data to track hygiene standards and improve sanitation services.
10) Research and Development: This system serves as a strong foundation for further research in smart waste management, water conservation, and environmental monitoring. It can be expanded into an integrated smart sanitation network that contributes to sustainable urban planning.

XIII. CONCLUSION & FUTURE SCOPE

The IoT and GSM Enabled Sanitation Management System successfully demonstrates how sensor integration and wireless communication can revolutionize sanitation monitoring. By combining solar power, ESP32 control, IoT connectivity, and GSM communication, the system provides a sustainable and intelligent approach to maintaining hygiene and environmental safety. The automatic detection of waste levels, gas leakage, and human presence ensures timely alerts and actions, minimizing health risks and maintenance costs. This solution can be effectively implemented in urban waste management, smart cities, and rural sanitation initiatives to promote cleaner and safer environments.

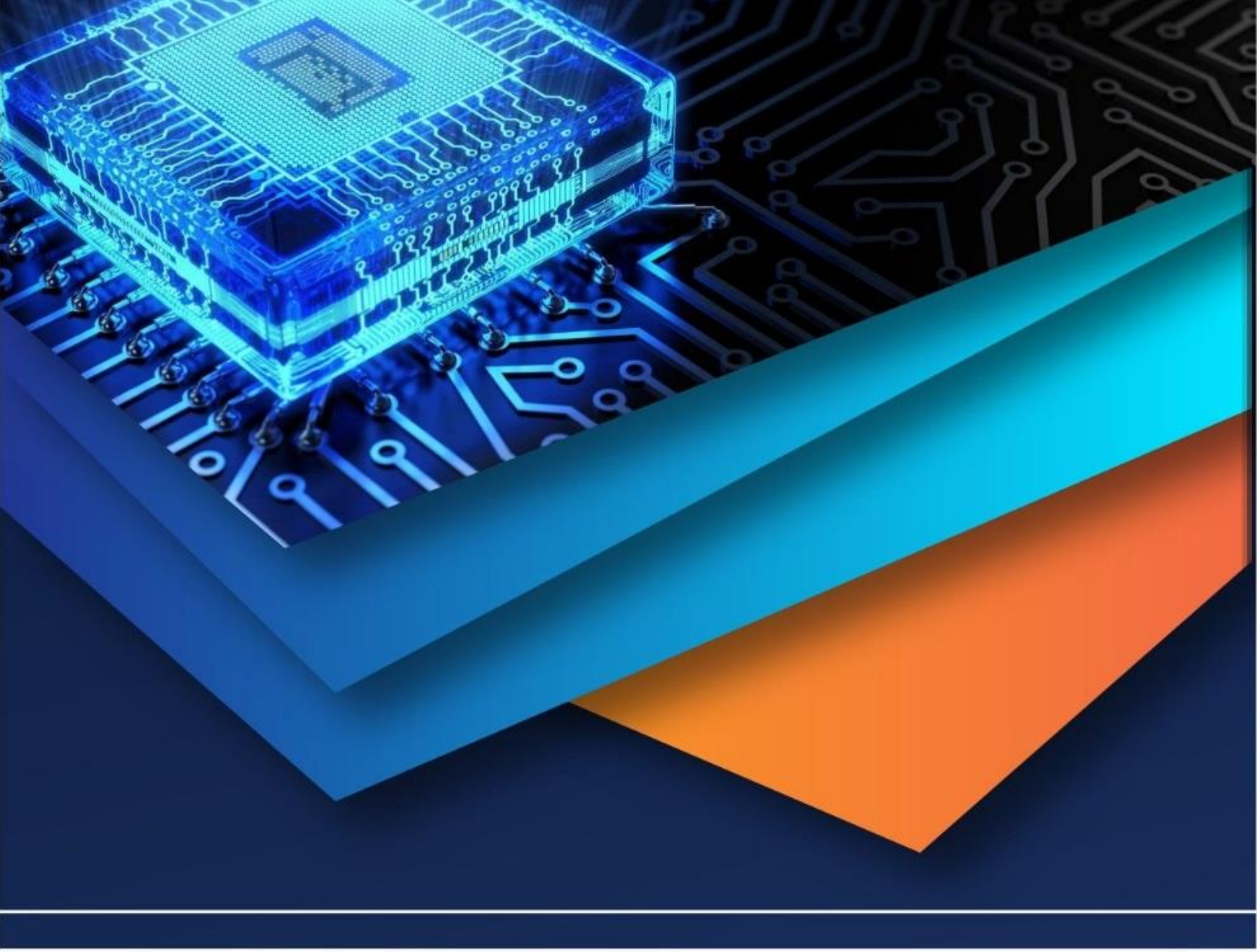
A. Future Scope

In the major project, an ultrasonic sensor will be integrated to detect the waste bin level, ensuring timely waste disposal and preventing overflow. The future scope of the system includes:

- Smart Waste Level Monitoring: Automatic detection of bin levels to alert authorities for efficient waste collection and management.
- Data Analytics and Prediction: Using collected data to predict cleaning schedules and optimize maintenance based on usage patterns.
- Dashboard Integration: Developing a mobile or web-based interface for real-time monitoring, alerts, and performance tracking of sanitation units.

REFERENCES

[1] Patel, K., & Modi, D. (2020). Smart Waste Management System Using IoT and GSM Technology. International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET), 9(4), 2345–2350.
[2] Kanchan, R., & Singh, R. (2019). IoT Based Smart Garbage Monitoring and Alert System Using Arduino and GSM. International Journal of Engineering Research & Technology (IJERT), 8(6), 512–516.
[3] Chaware, P. R., Raut, R. S., & Tembhurne, S. A. (2021). IoT Based Smart Waste Management System for Smart Cities. International Research Journal of Engineering and Technology (IRJET), 8(2), 205–210.
[4] Sathish, R., & Kumar, S. (2019). Smart Waste Management System Using IoT and Machine Learning. IEEE International Conference on Communication and Signal Processing (ICCSP), 1–5.
[5] ESP32 Technical Reference Manual, Espressif Systems, 2021. Available at: https://www.espressif.com
[6] Islam, M. N., & Rahman, M. M. (2020). IoT Based Waste Bin Monitoring System Using GSM and Ultrasonic Sensor. International Journal of Computer Applications, 176(29), 1–5.
[7] Singh, A., & Gupta, N. (2022). Solar Powered IoT Based Smart Garbage Monitoring System. International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering (IJAREEIE), 11(3), 1458–1464.
[8] Priya, S., & Rajesh, P. (2021). IoT Based Environmental Monitoring System Using ESP32. International Journal of Scientific Research in Engineering and Management (IJSREM), 5(8), 112–118.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY