Integrated Gas Analyser

Course Project for Embedded Systems Workshop
EC3.202, Monsoon 2024

Hardik Chadha[1,*], Arnav Sharma[2,*], Hiten Garg[3,*], Kushal Mangla[4,*], Prof. A. Shrivastava[5,+]
[1,2,3,4]*IIIT Hyderabad*, [5]*CVEST, IIIT Hyderabad*
[*]*Team Members*, [+]*Project Advisor*

***Abstract*—Toxic gas exposure in industrial environments, such as manufacturing and chemical processing, poses serious risks to worker safety. Existing gas detection systems are often bulky and lack real-time monitoring, limiting timely responses to hazards; or might be unaffordable for a majority of industrial sites. This project aims to develop a low-cost, portable gas analyzer that enhances safety by providing continuous, real-time detection of harmful gases.**

I. Introduction

Air quality monitoring is becoming increasingly important as environmental pollution, industrial emissions, and household gas leaks pose significant risks to health and safety. Harmful gases like carbon monoxide (CO), smoke, and liquefied petroleum gas (LPG) are major contributors to air pollution and indoor hazards. Real-time monitoring of these gases can help prevent life-threatening incidents by alerting individuals and authorities when the gas concentration reaches dangerous levels.

This paper presents the development of an integrated Gas Analyzer System that detects various harmful gases such as LPG, CO, and smoke using the MQ-2 sensor, and transmits the data wirelessly using LoRaWAN technology. The system is designed to operate in areas with low network coverage and utilizes IoT platforms to provide remote monitoring. This project showcases how embedded systems, sensor technologies, wireless communication, and cloud-based data analysis can be leveraged to create a robust environmental monitoring solution.

The paper details the hardware setup, sensor integration, data transmission via LoRa, cloud data storage using ThingSpeak, CockroachDB and PostgreSQL; and real-time notifications through Twilio, an API service for SMS notifications. We also demonstrate the development of a graphical user interface (GUI) using HTML and CSS to visualize the collected data, making it user-friendly and accessible.

II. Problem Statement

In various industries, toxic gas exposure presents significant health and safety risks to workers. Current solutions often fall short due to their bulkiness, lack of portability and insufficient real-time monitoring, and high costs.

To address these challenges, we aim to develop a low-cost portable gas analyser. The focus will be on making an architecture that can be used to collect and transfer the gas data and is both user-friendly and cost-effective.

The system should be capable of detecting hazardous gases in real-time and transmitting the data wirelessly over long distances. The system must be scalable to monitor multiple locations, offer reliable data visualization, and provide immediate notifications when gas concentrations exceed safety thresholds. Moreover, the solution must work effectively in areas with limited network connectivity, such as rural or industrial zones (mines etc.).

The objective of this project is to design and implement an IoT-based gas monitoring system that integrates gas sensors, long-range wireless communication (LoRaWAN), and cloud-based data storage. The system will offer real-time gas detection, remote monitoring, and automated notifications to ensure timely response to potential hazards, improving overall safety in both industrial and residential environments.

III. Objectives

Objectives for the project are as follows-

- **Real-time Detection of Hazardous Gases**: To design a system capable of detecting harmful gases such as LPG, carbon monoxide (CO), and smoke in real-time.
- **Long-range Wireless Communication**: To implement a reliable, low-power, long-range wireless communication for transmitting gas sensor data over distances of up to few hundred meters.
- **Cloud-based Data Storage and Visualization**: To enable users to monitor gas concentration levels remotely via a web-based graphical user interface (GUI).
- **Automated Threat Notifications**: To implement mechanism for sending real-time notifications to users when gas concentrations exceed predefined safety thresholds, ensuring timely response to potential hazards.
- **On-site Monitoring**: To provide a local monitoring solution using an LCD display that shows current gas levels on-site for immediate assessment.
- **Data Redundancy and Offline Access**: To ensure data reliability by storing sensor readings locally on an SD card in case of network connectivity loss, allowing periodic retrieval and analysis of historical data.
- **User-friendly Web Application**: To develop a web-based GUI that allows users to interact with the system, view

real-time data, analyze historical gas levels, and configure alerts, making the system accessible and easy to use.

- **Scalability for Multiple Locations**: To design the system to be scalable and capable of monitoring multiple sites, making it suitable for industrial, residential, and remote environments.

IV. Proposed Solution

Our solution comprises of the following components-

- **MQ-2 Gas Sensor for Gas Detection:** The system uses the MQ-2 gas sensor to detect the concentration of harmful gases such as LPG, CO, and smoke. The MQ-2 sensor provides reasonably accurate measurements of gas concentrations, suitable for most practical applications. This sensor can detect increase in concentrations of gases, allowing early detection of leaks or hazardous conditions while being reasonably cheap.
- **ESP-32 Microcontroller as Microcontroller:** An ESP-32 microcontroller serves as the central unit for the system. It collects data from the MQ-2 sensor and prepares it for transmission. The ESP-32 is an ideal choice due to its low power consumption, built-in Wi-Fi, and SPI communication capabilities, making it highly suitable for our usecase.
- **LoRaWAN for Long-range Wireless Communication:** To enable long-distance communication between the gas detection unit and the monitoring center, the system employs Ra-02 LoRaWAN modules. LoRa technology is chosen for its ability to transmit data over distances of up to few hundred meters with minimal power consumption. This ensures that gas levels can be monitored in real-time, even in remote or industrial environments with limited or no internet connectivity. Also, its range can be increased easily by introducing repeater modules.
- **Cloud-based Storage and Visualization using ThingSpeak and CockroachDB:** The collected gas data is transmitted to the ThingSpeak IoT platform, where it is stored and visualized in real-time. ThingSpeak provides a user-friendly dashboard that displays gas concentration levels in the form of graphs and charts. This allows users to analyze historical data and observe trends in gas levels, which can be critical for detecting recurring patterns or identifying potential hazards. The data from ThingSpeak is retrieved using Node.JS integrated with PostgreSQL. This data is uploaded to a cloud database created on CockroachDB, a distributed Database Management System. This online database can be used to create a user interface platform, enabling multiple users on our platform along with allowing users to view their sensor data securely.
- **Real-time Notifications via Twilio API:** To enhance safety, the system is integrated with Twilio's API to send real-time WhatsApp notifications to users when gas concentrations exceed predefined safety thresholds. This feature ensures immediate alerts are sent to responsible parties, enabling prompt action to prevent potential health and safety risks.
- **Local Monitoring via LCD Display:** In addition to remote monitoring, the system includes an on-site LCD display to show real-time gas concentration levels. This allows personnel near the gas detection unit to quickly assess the situation and take necessary actions without needing to access the web-based interface.
- **SD Card for Local Data Storage:** To ensure data reliability, especially in areas with intermittent network connectivity, the system includes an SD card module for local storage of sensor readings. This provides data redundancy and allows for the retrieval and analysis of historical data in case of network or cloud service failures.
- **User-friendly Web Application for Remote Monitoring:** User-friendly Web Application for Remote Monitoring: Our motive is to create a application that allows users to remotely monitor the system from any internet-connected device. So that it provides an intuitive platform to track real-time gas levels, set customizable alert thresholds, and access historical data. Creating a system which is both accessible and effective for monitoring and alert configuration.

V. System Architecture

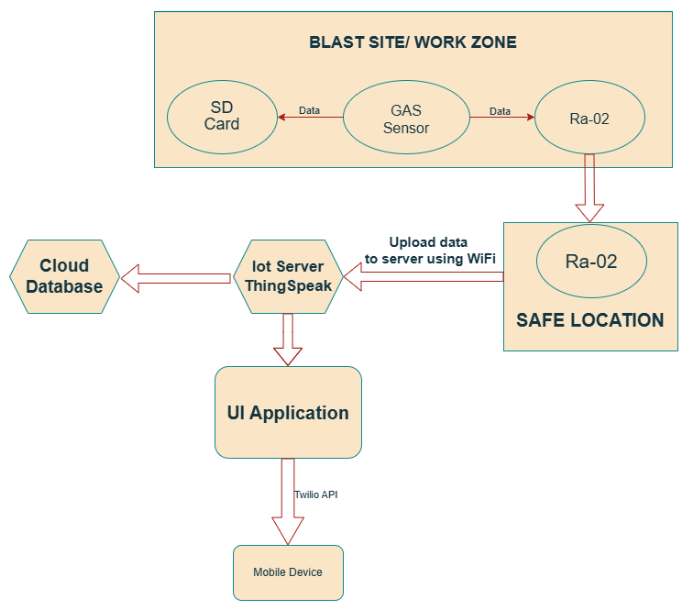

Fig. 1: System Architecture Diagram

The system architecture consists of multiple hardware and software components working together to ensure seamless data acquisition, transmission, processing, and visualization.

A. Hardware Components

The hardware setup includes sensors, a microcontroller, and communication modules:

- **ESP-32 Microcontroller**: In this setup, two ESP32 modules are used: one as the main node (sender module) paired with sensor, SD card and LCD, and the other as the receiver for the LoRa mechanism. The code for these is present here: Receiver [1] and Sender [1].
- **MQ-2 Gas Sensor**: One MQ-2 sensor is used for gas detection in our implementation, though the architecture allows the sensor change to more advanced and precise sensors with minimal changes in the code.
- **LoRaWAN (Ra-02) Module**: Two Ra-02 LoRa modules are used for transmitting data over long distances using low-power, long-range wireless communication. One module is at the sender node to transmit sensor data to a receiver node where the other module is located.

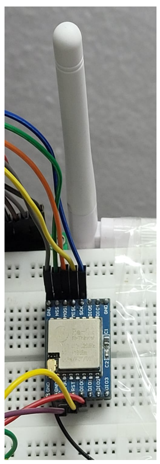

Fig. 2: LoRa Module

- **LCD Display**: A small LCD display is attached to the system to provide real-time, on-site monitoring. The display shows current gas concentrations, and displays whether the conditions are safe or not.

Fig. 3: LCD Display

- **SD Card Module and SD Card**: To ensure data persistence, the system is equipped with an SD card module [5], which helps writing the data to the SD card. Also a 32 GB SD card formatted in Fat32 file system is used to store the data. The SPI [4] pins of the ESP32 are used for communicating with the SD card module.

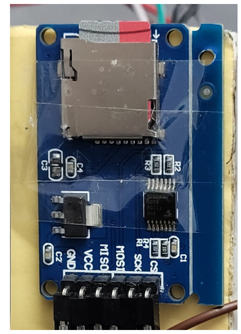

Fig. 4: SD Module

B. Cloud and Data Management

The system leverages cloud services for real-time data visualization, storage, and alerts:

- **CockroachDB:** An SQL database is integrated into the system with the help of CockroachDB, providing a distributed, scalable and resilient database solution for storing data required in our solution. The handling of concurrent requests, and upholding of ACID [1] characteristics for the database make CockroachDB ideal for use in an IoT application such as ours [7]. Initial schema of the database used is shown in the image below. The database can be scaled as per the needs of the customer.

[2] - Beyond Infinity @ beyond-infinity-esw:gas_analyser
Success

show columns from sensor_data;

Export results
Executed in 44 ms

	column_name	data_type	is_nullable	column_default	generation_expression	indices	is_hidden
1	field1	FLOAT8	true	null		["sensor_data_pkey"]	false
2	field2	FLOAT8	true	null		["sensor_data_pkey"]	false
3	field3	FLOAT8	true	null		["sensor_data_pkey"]	false
4	created_at	TIMESTAMPTZ	true	current_timestamp()		["sensor_data_pkey"]	false
5	rowid	INT8	false	unique_rowid()		["sensor_data_pkey"]	true

Fig. 5: DB Schema used

- **ThingSpeak IoT Platform** [3]: After data is transmitted to the receiver node, it is uploaded to ThingSpeak. It provides a comprehensive environment for real-time data collection and visualization. Data is presented as time-series graphs, allowing users to track gas concentration trends over time. It has built-in analysis tools help in identifying patterns.
- **Twilio API Integration**: The system is integrated with Twilio to send real-time WhatsApp alerts to users when gas concentrations exceed safety thresholds. This feature ensures immediate notification in case of dangerous gas levels, enhancing the overall safety of the system.

[1] Atomicity, Consistency, Isolation and Durability properties ensure reliability of a database in transactions.

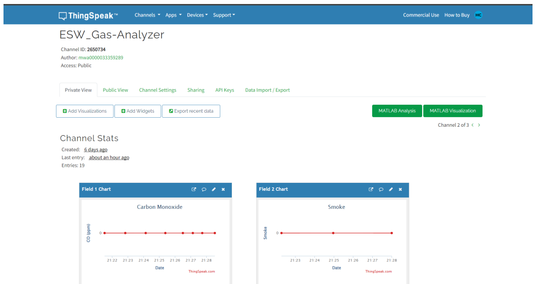

Fig. 6: Thingspeak Server

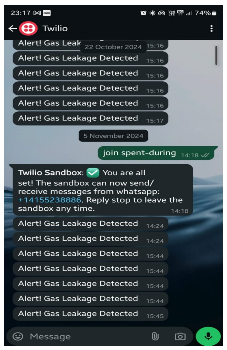

Fig. 7: Twilio API

C. User Interface

A web-based GUI is developed to provide users with a comprehensive view of the system's operation:

- **Web Application [6]**: A web-based graphical user interface (GUI) was developed using Node.js, Express.js, HTML, and CSS, offering a user-friendly interface for accessing sensor data remotely. Users can view real-time gas concentration levels and access historical data. It stores the gas data in one database and the registered users in another database.The employees of the company can register and login to view the real time data of the sites where the model is deployed.

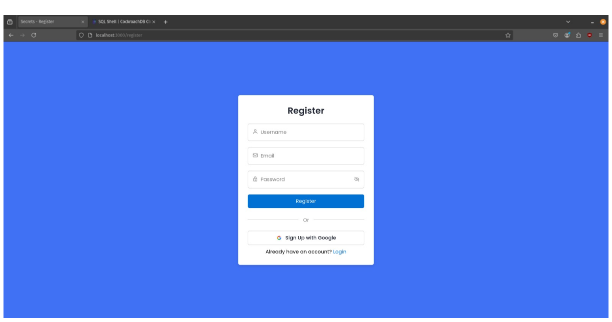

Fig. 8: Website Register Page

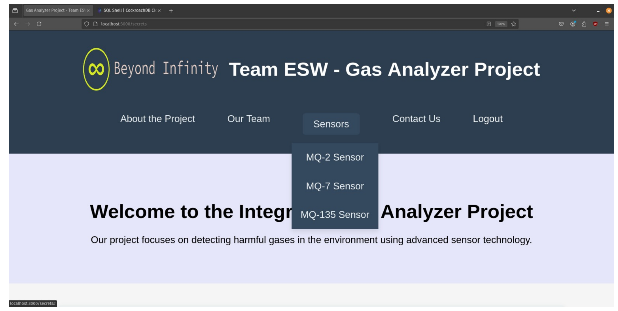

Fig. 9: Website Home Page

VI. PCB Design

We designed and fabricated Printed Circuit Boards for the sender and receiver modules of our solution using Autodesk Eagle.

A. Sender Module

The images for the sender module's circuit diagrams and board diagrams can be seen below. The files can be accessed through the GitHub repository. [8]

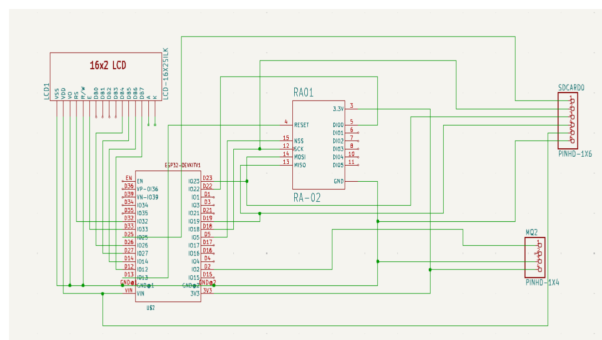

Fig. 10: Sender Circuit Diagram

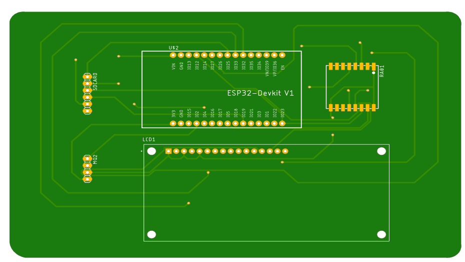

Fig. 11: Sender Board(Top View)

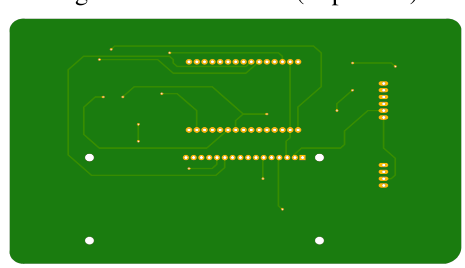

Fig. 12: Sender Board(Bottom View)

B. Receiver Module

The images for the receiver module's circuit diagram and board diagrams can be seen below. The files can be accessed through the GitHub repository. [8]

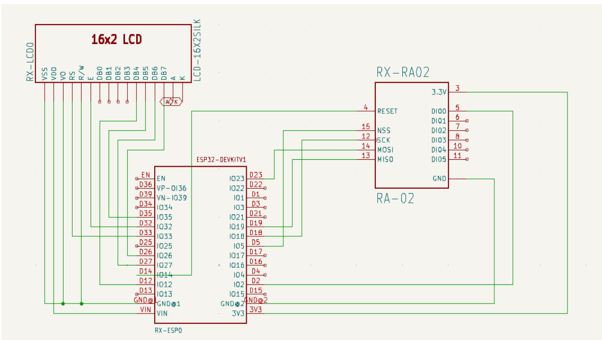

Fig. 13: Sender Circuit Diagram

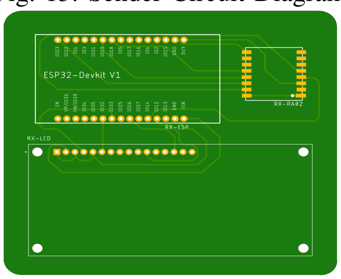

Fig. 14: Receiver Board(Top View)

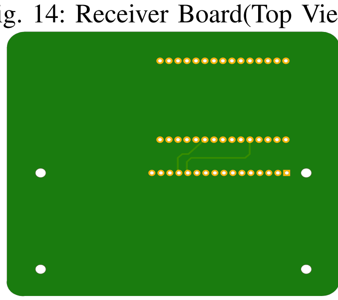

Fig. 15: Receiver Board(Bottom View)

VII. 3D Design

We designed the 3D design for the sender and receiver modules of our solution using Fusion 360. The link to all files can be found here : [9]

A. Sender Module

The images for the sender 3D design can be seen below. The files can be accessed via the GitHub repository.

B. Receiver Module

The images for the receiver 3D design can be seen below. The files can be accessed through the GitHub repository.

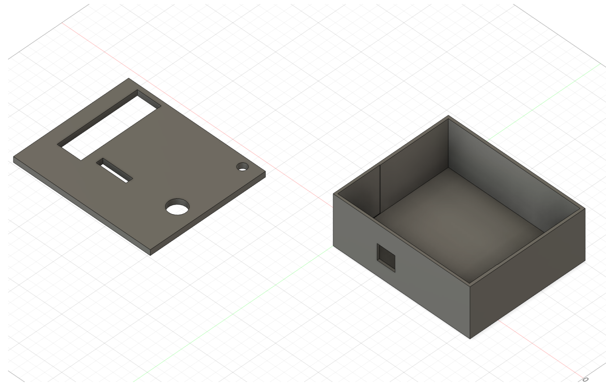

Fig. 16: Sender Module

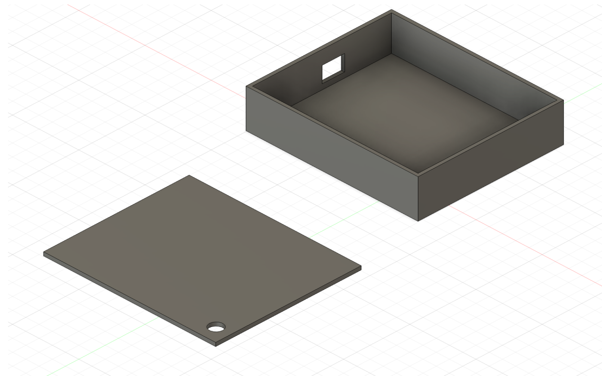

Fig. 17: Receiver Module

VIII. LoRa Analysis

To evaluate the performance and reliability of the LoRaWAN communication system, we conducted a some experiments over varying distances. The purpose of these tests was to measure the strength of the LoRa signal(RSSI), assess packet transmission quality(SNR), and analyze the effect of obstacles (such as walls and floors) on signal integrity.

A. Testing LoRa Values Over a Distance of 1-5 Meters

In this experiment, we tested the LoRa signal strength and packet transmission performance over a short distance of 1-5 meters in an open area. The purpose was to establish baseline performance in an ideal, interference-free environment.

- **Setup**: The LoRa transmitter was placed at one end of the room, and the receiver was moved to distances ranging from 1 meter to 5 meters.
- **Parameters measured**: Signal-to-Noise Ratio (SNR), Received Signal Strength Indicator (RSSI), and packet delivery success.
- **Results**: The signal strength was optimal, with RSSI values remaining between -30 dBm and -40 dBm. Packet transmission success was near 100%, with minimal packet loss and high SNR values ranging from 8 to 10 dB.

B. Same Floor Experiment (5-30 Meters)

We then extended the test distance to 5-30 meters on the same floor. This experiment was designed to assess LoRa's performance in a confined indoor environment with potential interference from furniture and walls.

- **Setup**: Both the transmitter and receiver were placed on the same floor, with the distance increased gradually to 30 meters. Obstacles like furniture were present in the room.
- **Parameters measured**: RSSI, SNR, and packet success rate.
- **Results**: RSSI values dropped slightly, ranging from -45 dBm to -60 dBm as the distance increased. The SNR decreased down to -2dB on average due to rooms in between and also the increasing distance, and packet loss began to occur, especially at the 30-meter mark.

C. Different Floor Experiment (1-2 Floors)

In this test, we analysed LoRa performance between different floors of the same building. This experiment introduced structural interference, such as concrete ceilings and walls, which are common in multistory buildings.

- **Setup**: The transmitter was placed on the third floor, while the receiver was taken to the first, second, and fourth floors.
- **Parameters measured**: RSSI, SNR, and packet success rate.
- **Results**: The signal degraded more significantly, with RSSI values dropping to -70 dBm on the first floor and -80 dBm on the second floor. The SNR values observed a very pattern-like behaviour that is, when the distance was increased until the stairs then the SNR value decreased and when the receiver was just a floor below then it increased due to less obstacles between them. This change in value was observed for each descending floor. Packet loss increased significantly in this case.

D. Long Distance

For the final experiment, we tested LoRa communication between two different buildings to evaluate signal penetration through obstacles like trees, people and across open spaces between the buildings.

- **Setup**: The transmitter was placed in one building, while the receiver was taken to an adjacent building separated by a distance of approximately 30-75 meters.
- **Parameters measured**: RSSI, SNR, and packet success rate.
- **Results**: The RSSI values dropped significantly, falling to -90 dBm at 30 meters and -100 dBm at 30 meters. The SNR dropped significantly from 5 to -8 dB, and the packet success rates fell. The signal strength reduced significantly with the distance increased.

E. Analysis Summary

The experiments demonstrate that LoRa performs exceptionally well in short-range, open environments with minimal interference. However, as obstacles such as walls, floors, and the distance between buildings increase, signal strength and reliability diminish significantly. While LoRa is effective for long-range communication in rural and less obstructed areas, its performance in dense urban environments or multi-story buildings may require the use of repeaters or stronger signal antennas to ensure reliable communication.

IX. Cost Analysis

The cost analysis of the Gas Analyzer project is divided into hardware costs and software/infrastructure costs. All values are provided in approximate INR.

A. Hardware Costs

The hardware components include sensors, microcontrollers, communication modules, and other peripherals. Below is an estimated breakdown of the costs in INR:

Component	Quantity	Unit Cost (INR)	Total Cost (INR)
ESP-32 Module	2	500	1,000
MQ-2 Gas Sensor	1	150	150
Ra-02 LoRa Module	2	500	1,000
SD Card Module	1	100	100
16 GB SD Card	1	400	400
LCD Display	1	200	200
Other Connectors	-	100	100

TABLE I: Hardware Costs in INR

Total Hardware Cost (Approx.): Rs. 3,000

B. PCB Cost

The cost of fabrication of the custom PCB that we have created for the Sender and Receiver.

PCB Fabrication Cost: Rs. 3,500

C. Software/Infrastructure Costs

When the system is deployed in a real-world scenario, there will be associated software and infrastructure costs. These may include expenses for cloud services, IoT platform subscriptions, data storage, and maintenance of the software components necessary for seamless operation.

D. Total Estimated Project Cost

The total estimated cost of the Gas Analyzer project, including hardware and PCB costs only, is provided below:

Total Project Cost (Approx.): Rs. 6,500

X. Modular Hardware Design

One of the key strengths of our Gas Analyzer project is the modular design of the hardware. The system is structured in such a way that the MQ-2 sensor, which is primarily used for detecting LPG, smoke, and carbon monoxide, can be easily replaced with other gas sensors with minimal modifications to the code.

A. Flexible Sensor Integration

The hardware architecture supports a variety of gas sensors due to the uniform pin configurations and standard interfacing techniques employed. This design allows for seamless integration of sensors such as MQ-7 annd MQ-135 without the need for significant hardware modifications.

B. Minimal Code Changes

The flexibility in the design is complemented by the software, where only minor adjustments to the sensor calibration parameters and data reading logic are required when switching sensors. This modular approach reduces the complexity of reconfiguring the system for different gases, making it highly adaptable to various environmental monitoring needs.

For instance, replacing the MQ-2 sensor with a new sensor involves:

- Updating the sensor input methods in the Sender Module code according to the new sensor library.
- Adjusting the gas concentration thresholds specific to gases the new sensor measures.
- If the new gases are not already in the system, tables related to their data needs to be changed in the system.

This approach ensures that the system can be repurposed for a wide range of applications with minimal effort, increasing its utility in both industrial and residential settings.

C. Future Expansion

This project demonstrates a functional implementation of an IoT system using ESP32 modules with LoRa communication. However, there are several areas for potential improvement and future enhancements:

1) **Sensor Calibration:** Adding algorithms for better sensor calibration than the library available calibration mechanism to enhance system accuracy in detecting the gas concentrations.
2) **Usage of Better Sensors:** Explore the integration of higher-precision or industrial-grade sensors to improve the accuracy and reliability of the data collected, ensuring better performance in critical applications, though this will come at cost of increasing the price of the device.
3) **Interfacing with Additional Sensors:** Extend the system to interface with other types of sensors, such as environmental sensors (temperature, humidity) or industrial sensors (vibration, pressure) for diversified applications.
4) **Power Optimization:** Investigate techniques to optimize power consumption in the ESP32 modules without compromising real-time data transmission. Solutions could include efficient scheduling of operations or the use of low-power components where feasible.
5) **Integration with Advanced IoT Platforms:** Extend the system by integrating it with advanced IoT platforms like AWS IoT Core, Microsoft Azure, or Google Cloud IoT for enhanced data processing, analytics, and visualization.
6) **Multi-Node LoRa Network:** Expand the system to support multiple sender nodes communicating with a single receiver, enabling the development of a larger sensor network for broader applications.

XI. Summary

The Integrated Gas Analyzer is an IoT-based solution designed for real-time monitoring of harmful gases such as LPG, carbon monoxide (CO), and smoke. Utilizing an ESP-32 microcontroller and the MQ-2 sensor, the system collects gas concentration data and transmits it using LoRaWAN modules for long-range, low-power communication. The data is visualized on a web-based GUI and stored on cloud platforms like ThingSpeak and CockroachDB, ensuring remote access and historical analysis. Additionally, the system incorporates Twilio API for real-time WhatsApp notifications and an LCD display for on-site monitoring. Experiments demonstrated LoRa's effective performance in short-range environments but noted signal degradation over longer distances and through obstacles. The project's modular design allows for easy sensor replacement with minimal code changes, making it scalable and adaptable to various applications. With a total cost of approximately Rs. 6,500, the system combines affordability, reliability, and user-friendliness, offering a practical solution for industrial and residential safety monitoring.

XII. Personal Contributions

- **Hardik Chadha (2023111031):**
  1) Integrated the SD card module with the Sender module.
  2) Integrated the LCD display with the Sender Module.
  3) Wrote the code for integrating SD card and LoRa module on the same SPI pins of the sender module ESP and debugging the issues faced in these parts.
  4) Helped in the inititial testing of MQ-2 Sensor.
  5) Helped in the testing of LoRa Modules.
  6) Helped in the data collection for different experiments for LoRa.
  7) Assisted in the Soldering of the PCB connections.
- **Arnav Sharma (2023111033):**
  1) Integrated HTML with ThingSpeak, to enable live retrieval of data from ThingSpeak to be displayed on the web-page.
  2) Used NodeJS and PostgreSQL to connect server, ThingSpeak and CockroachDB.
  3) Created database/backend on CockroachDB Cloud for storing data.
  4) Made the circuit schematic to be used in the PCB Design.
  5) Designed the PCB using Autodesk Eagle.
  6) Helped in collection of data for LoRa analysis.
  7) Made the PPT's for progress of each week, as well as for the final presentation.
- **Kushal Mangla (2023101026):**
  1) Wrote the codes for Lora Modules.

2) Coding and implementation of TWILIO API with Node and Express JS.
3) Made the backend for the the user interface of our website, implementing nodejs, expressjs, postgress.
4) Intergrating the EJS files with our backend server.
5) Helped in hardware integration with Hiten.
6) Helped in data collection for the lora analysis.
7) Analysed the data using python scripts and made graphs.
8) Made the 3D design for both the sender and receiver modules.

- **Hiten Garg (2023101116):**

1) Analyzed the codes along with Kushal.
2) Implementation of messaging system using TWILIO API with whatsapp integration.
3) Made the HTML page for the website to showcase the overview of our project.
4) Implemented the hardware like connections for both the sender and receiver modules.
5) Worked on data collection for different experiments.
6) Analysed the data using python scripts and made graphs.
7) Made the 3D design for both the sender and receiver modules.

REFERENCES

[1] Lora Reciever

[2] Lora Sender

[3] Thingspeak Channel

[4] Resource for SPI

[5] SD Card adapter Interfacing

[6] Complete GUI Code

[7] CockroachDB Features

[8] PCB Files

[9] 3D Designs