# IoT-BASED SMART BIN

# A

# MAJOR PROJECT-II REPORT

Submitted in partial fulfillment of the requirements

for the degree of

**BACHELOR OF TECHNOLOGY**

in

**COMPUTER SCIENCE & ENGINEERING**

By

**GROUP NO. 08**


| | |
|---|---|
| **Abhay Prince** | **0187CS211003** |
| **Amit Kumar** | **0187CS211022** |
| **Amit Kumar** | **0187CS211023** |
| **Ankit Kishor** | **0187CS211029** |


Under the guidance of

**Prof. Mayank Kurchaniya**

(Assistant Professor)



**Department of Computer Science & Engineering**
**Sagar Institute of Science & Technology (SISTec), Bhopal (M.P.)**

**Approved by AICTE, New Delhi & Govt. of M.P.**
**Affiliated to Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal (M.P.)**


**June -2025**

**Sagar Institute of Science & Technology (SISTec), Bhopal (M.P)**

**Department of Computer Science & Engineering**


***CERTIFICATE***

We hereby certify that the work which is being presented in the B.Tech. Major Project-II Report entitled **IoT BASED SMART BIN,** in partial fulfillment of the requirements for the award of the degree of ***Bachelor of Technology,*** submitted to the Department of **Computer Science & Engineering**, Sagar Institute of Science & Technology (SISTec)**,** Bhopal (M.P.) is an authentic record of our own work carried out during the period from Jan-2025 to June-2025 under the supervision of **Prof. Mayank Kurchaniya.**

The content presented in this project has not been submitted by us for the award of any other degree elsewhere.

| **Abhay Prince** | **Amit Kumar** | **Amit Kumar** | **Ankit Kishor** |
|---|---|---|---|
| **0187CS211003** | **0187CS211022** | **0187CS211023** | **0187CS211029** |

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

***Date:***

| **Prof. Mayank Kurchaniya**<br>**Project Guide** | **Dr. Amit Kumar Mishra**<br>**HOD, CSE** | **Dr. D.K. Rajoriya**<br>**Principal** |
|---|---|---|

# ACKNOWLEDGEMENT

We would like to express our sincere thanks to **Dr. D. K. Rajoriya**, **Principal, SISTec** and **Dr. Swati Saxena, Vice Principal SISTec,** Gandhi Nagar, Bhopal for giving us an opportunity to undertake this project.

We also take this opportunity to express a deep sense of gratitude to **Dr. Amit Kumar Mishra, HOD, Department of Computer Science & Engineering** for his kindhearted support

We extend our sincere and heartfelt thanks to our guide, **Prof. Mayank Kurchaniya,** for providing us with the right guidance and advice at crucial junctures and for showing us the right way.

I am thankful to the **Project Coordinator, Prof. Deepti Jain,** who devoted her precious time in giving us the information about various aspects and gave support and guidance at every point of time.

# TABLE OF CONTENTS

# ABSTRACT

This project is a part of the Smart City initiative, leveraging IoT technologies to enhance urban waste management systems. In a Smart City, internet connectivity is ubiquitous, enabling seamless communication between devices and centralized platforms. The Smart Bin system is designed to optimize waste collection processes by prioritizing which bins should be emptied first. This prioritization is achieved by transmitting real-time data to the ThingSpeak platform, where the Municipal Corporation can access graphical representations of bin statuses and make informed decisions. The Smart Bin is equipped with an ultrasonic distance sensor to monitor the fill level of waste in the bin. This sensor measures the distance between the top of the bin and the waste, providing accurate data on how full the bin is. Additionally, a DHT11 sensor is integrated to measure temperature and humidity levels within the bin. This data is crucial for assessing moisture levels, which can help in identifying bins that may require urgent attention to prevent odor or bacterial growth. Furthermore, a PIR (Passive Infrared) sensor is used to detect the presence of living beings inside the bin, ensuring safety and preventing potential hazards. The data collected from these sensors is wirelessly transmitted to the ThingSpeak platform, where it is visualized through graphs and charts. This allows the Municipal Corporation to prioritize waste collection routes efficiently, reducing fuel consumption, labor costs, and environmental impact. By focusing on bins that are full or have critical conditions, the system ensures timely waste collection and maintains cleanliness in the city. In the future, the system can be enhanced by integrating AI models to further optimize waste management. AI algorithms can analyze historical data to predict fill patterns, enabling proactive scheduling of waste collection. Additionally, machine learning can be used to classify waste types, facilitating recycling efforts and promoting sustainability. The integration of AI would make the Smart Bin system even more intelligent, adaptive, and efficient.

# LIST OF ABBREVIATIONS

| ACRONYM | FULL FORM |
|---|---|
| IoT | Internet of Things |
| UDS | UltraSonic Distance Sensor |
| SB | Smart Bins |
| PIR | Passive InfraRed |
| IDE | Integrated Development Environment |
| SRS | Software Requirements Specifications |
| DFD | Data Flow Diagram |
| RTC | Real Time Clock |

# LIST OF FIGURES

# LIST OF TABLES

# Chapter 1
# Introduction

# CHAPTER-1
# INTRODUCTION

## 1.1 About Project

Urban waste management faces significant challenges, leading to overflowing bins, unsanitary conditions, and environmental hazards. Our IoT-Based Smart Bin is designed to tackle these issues through real-time monitoring and data-driven waste collection. It incorporates an ESP32 microcontroller with ultrasonic, PIR, and DHT11 sensors to measure garbage levels, detect human presence, and monitor temperature and humidity. The system features red, yellow, and green indicator lights, allowing passersby to visually assess the bin's status and notify municipal authorities when needed. All sensor data is transmitted to the ThingSpeak cloud platform, enabling waste management teams to analyze trends, prioritize cleaning schedules, and enhance efficiency. This solution helps in reducing waste overflow, minimizing mosquito breeding, and maintaining hygiene, especially during festivals and high-waste periods. By leveraging IoT technology, the Smart Bin ensures efficient waste disposal, improved urban cleanliness, and a step towards smarter, more sustainable cities.

## 1.2 Importance of Smart Waste Management

The IoT-Based Smart Waste Management System enhances urban cleanliness by monitoring and managing waste disposal efficiently. It uses ultrasonic, PIR, and DHT11 sensors to track garbage levels, human presence, and environmental conditions, transmitting data to ThingSpeak for analysis. This system helps reduce overflow, prevent mosquito breeding, and optimize waste collection schedules, ensuring better sanitation, cost efficiency, and a cleaner, smarter urban environment.

## 1.3 Mosquito Breeding: A Growing Concern

Mosquito-borne diseases pose a significant public health threat worldwide, contributing to illnesses such as dengue, malaria, and chikungunya. In India, the number of dengue cases has surged in recent years, with the National Vector Borne Disease Control Programme (NVBDCP) reporting over 200,000 cases annually. Stagnant waste and overflowing garbage bins create ideal breeding grounds for mosquitoes, increasing infection risks. Implementing smart waste management solutions can help eliminate these breeding sites, significantly reducing mosquito-borne disease outbreaks and improving public health and sanitation.

## 1.4 The Role of IoT in Smart Cities

IoT platforms such as ThingSpeak and AWS IoT enable real-time monitoring and management of urban infrastructure by collecting and analyzing data from connected sensors and devices. These platforms allow municipal authorities to optimize waste management, traffic control, energy consumption, and public safety. For example, IoT-based smart bins use sensors to detect garbage levels and transmit data to cloud servers, enabling efficient waste collection. Similarly, smart traffic systems analyze congestion patterns to improve road management. By integrating IoT technology, smart cities enhance efficiency, sustainability, and overall quality of urban life.

## 1.5 Current Challenges in Patient Monitoring Systems

### 1.5.1 Existing Systems

Many traditional waste management systems rely on manual monitoring and static collection schedules, leading to inefficiencies and hygiene issues. These systems face several limitations:

- **Irregular waste collection**, causing overflowing bins.
- **Lack of real-time monitoring**, making it difficult to optimize collection routes.
- **No automated alerts**, requiring manual inspections.

**1. Barriers in Implementation**

Real-time waste management systems face challenges in terms of **connectivity, cost, accuracy, and usability**:

**Network Availability**: Continuous internet connectivity is required for real-time data transmission, which can be unreliable in some areas.

**Sensor Efficiency**: Dust, moisture, and harsh weather conditions can impact sensor accuracy, requiring frequent calibration and maintenance.

**Cost & Scalability**: Implementing IoT-based waste management on a large scale requires high initial investment in infrastructure and cloud services.

**User Awareness**: Municipal authorities and the public may need training to understand and effectively utilize smart waste monitoring systems.

**Security & Data Privacy**: IoT devices are vulnerable to cyber threats, necessitating secure communication protocols to prevent data breaches.

## 1.6 Advantages of IoT-Based Smart Bin Systems

**Real-Time Waste Monitoring**: Authorities can track garbage levels in real time, ensuring timely waste collection and preventing bin overflow.

**Optimized Waste Collection**: Smart bins help waste management teams prioritize collection routes, reducing unnecessary fuel consumption and operational costs.

**Improved Urban Hygiene**: By preventing waste accumulation, smart bins help reduce foul odors, pest infestations, and mosquito breeding, leading to a healthier environment.

**Cost-Effective Waste Management:** Automating waste monitoring minimizes manual inspections, saving labor costs and improving efficiency.

**Enhanced Public Awareness**: Indicator lights (red, yellow, green) notify passersby and authorities about waste levels, encouraging community participation in waste management.

## 1.7 Applications of IoT-Based Smart Bin

### 1.7.1 Waste Level Monitoring

The IoT-based Smart Bin continuously monitors garbage levels using ultrasonic sensors, ensuring timely waste disposal and preventing bin overflow in public areas.

### 1.7.2 Elderly Care

Municipal corporations can use real-time data to optimize waste collection schedules, reducing operational costs and maintaining a cleaner city environment.

### 1.7.3 Public Health & Hygiene

By preventing waste accumulation, smart bins help reduce the spread of diseases, minimize mosquito breeding, and improve sanitation, especially in crowded areas.

### 1.7.4 Event & Festival Waste Management

During large public gatherings and festivals, smart bins help manage increased waste efficiently by providing real-time alerts, ensuring quick cleanup and reducing environmental hazards.

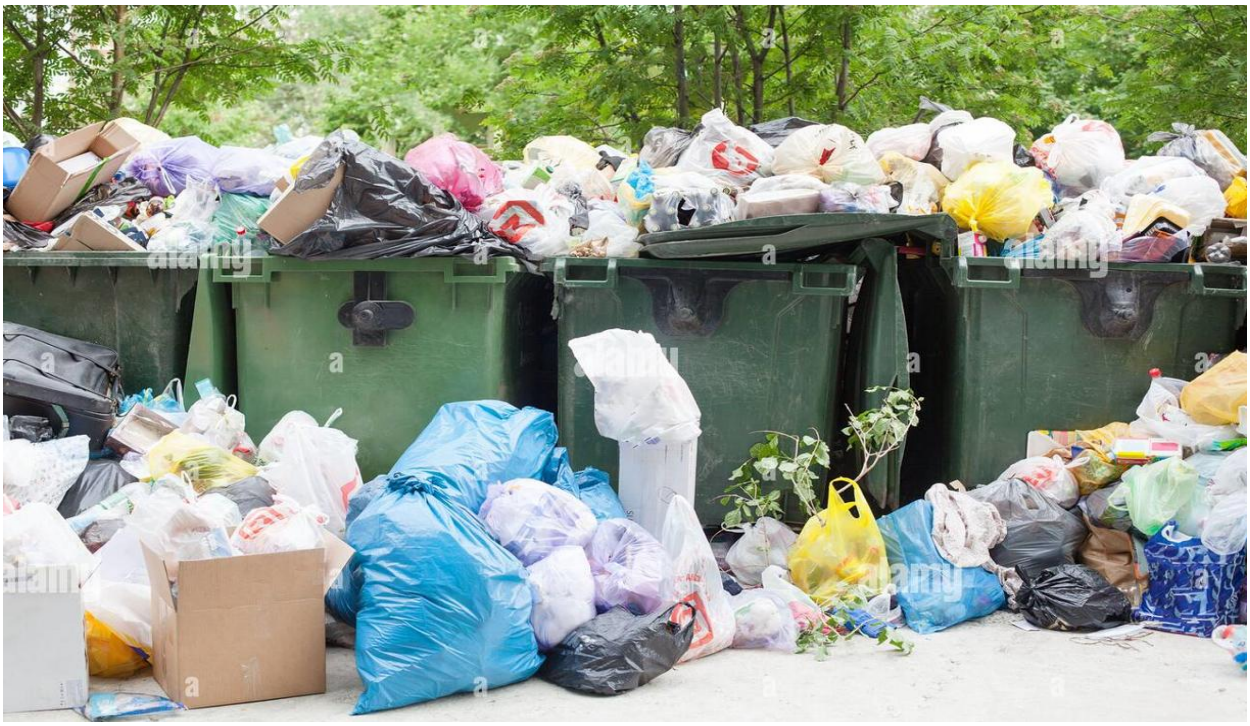

**Figure 1.1: Earlier Problem**

Figure 1.1 illustrates the earlier problem of waste management in urban areas, where traditional bins often overflow due to inefficient garbage collection systems. The image shows a scenario where waste is piled up beyond the bin's capacity, leading to unsanitary conditions, unpleasant odors, and environmental hazards. The accumulation of garbage attracts pests, promotes mosquito breeding, and increases the risk of disease outbreaks. Additionally, improper waste disposal affects the aesthetics of public spaces and contributes to pollution. The primary reason for such conditions is the lack of real-time monitoring and timely intervention by municipal authorities. Without an automated system, waste collection remains inefficient, resulting in delayed cleanups and excessive waste accumulation. Implementing IoT-based Smart Bins can help mitigate these issues by providing real-time data on garbage levels, allowing authorities to optimize waste collection schedules and maintain urban cleanliness effectively.

# Chapter 2
# Software & hardware requirements

# CHAPTER-2
# SOFTWARE AND HARDWARE REQUIREMENT

Software and hardware requirements include the essential tools, technologies, and physical components needed for a system to function efficiently. Software requirements involve operating systems and applications, while hardware requirements encompass devices like processors, memory, and storage.

## 2.1 SOFTWARE REQUIREMENTS

### 2.1.1 Arduino IDE: The Integrated Development Environment

The Arduino IDE is an open-source platform that provides an integrated development environment for writing, compiling, and uploading code to Arduino-compatible boards, including the ESP32. Its user-friendly interface and extensive community support make it an ideal choice for both beginners and experienced developers working on IoT projects.

**Key Features of Arduino IDE:**

- **Cross-Platform Compatibility:** The Arduino IDE is available for Windows, macOS, and Linux, allowing developers to work on their projects regardless of their operating system.
- **Simplified Code Writing:** The IDE supports C/C++ programming languages with simplified syntax, making it easier to write code for various applications.
- **Library Management:** Users can easily manage libraries needed for specific functionalities, such as RFID communication or Wi-Fi connectivity.
- **Serial Monitor:** The built-in serial monitor allows developers to view output from their code in real-time, facilitating debugging and testing processes.

To set up the Arduino IDE for the Track Access System, users must first download and install the software from the official Arduino website. Once installed, they can configure the IDE to support the ESP32 board by adding the appropriate board manager URLs and selecting the desired board type.

### 2.1.2 ThingSpeak

ThingSpeak is an IoT analytics service that allows you to aggregate, visualize, and analyze live data streams in the cloud. ThingSpeak provides instant visualizations of data posted by your devices to ThingSpeak.

### 2.1.3 Telegram Messaging

It can integrate Telegram with ESP32 or other microcontroller boards to send notifications, alerts, or control messages. This is useful for remote monitoring, automated alerts, and controlling IoT devices.

## 2.2 HARDWARE REQUIREMENTS

### 2.2.1 ESP32 Board

The ESP32 is a popular microcontroller developed by Espressif, commonly used in Internet of Things (IoT) applications. It offers a rich set of features and support for various tasks. Here's a breakdown of the key features of the ESP32 board:

**Key Features of ESP32 Board Support**

- Dual-core processor running at up to 240 MHz for high performance.
- Built-in Wi-Fi and Bluetooth (Classic and BLE) for wireless communication.
- 34 GPIO pins supporting digital, analog, PWM, I2C, SPI, UART, and more.
- 520 KB SRAM and up to 16 MB Flash memory for storing applications.
- Multiple power-saving modes for battery-powered projects.
- Supports OTA (Over-the-Air) updates for remote firmware updates.
- Ideal for IoT, home automation, wearables, and sensor networks.

**Figure 2.1: ESP32**

### 2.2.2 PIR Sensor

The Passive Infrared (PIR) sensor is a motion detection device that identifies the presence of living beings by sensing infrared radiation emitted from their bodies. It is widely used in automation and security systems to detect movement without direct contact.

**Key Features:**

- **Motion Detection:** The PIR sensor detects heat signatures from humans and animals, triggering actions based on movement.
- **Energy Efficient:** It consumes minimal power, making it suitable for battery-operated and IoT-based systems.
- **Wide Detection Range:** Most PIR sensors offer a detection range of 5–10 meters, making them effective for monitoring large areas.
- **Fast Response Time:** The sensor provides instant detection, ensuring real-time alerting and automation.

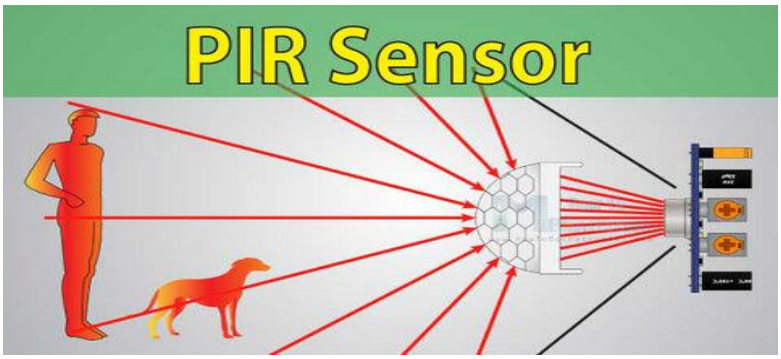


**Figure 2.2: PIR Sensor**

### 2.2.3 Connectors (Wires or Breadboard)

Wires are simple conductive materials that are used to establish a physical connection between two or more electronic components. A breadboard, on the other hand, is a tool used to build and test electronic circuits. It allows you to quickly and easily prototype a circuit without the need for soldering or special tools.

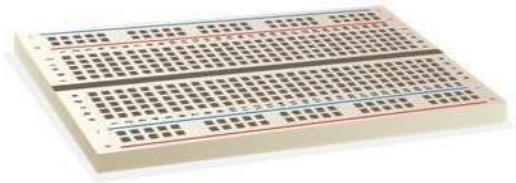

**Figure 2.3: Breadboard**

### 2.2.4 DHT11

The DHT11 sensor is a low-cost digital temperature and humidity sensor, widely used in simple projects where moderate accuracy is sufficient. It's a popular choice for makers, hobbyists, and beginners due to its affordability and ease of use.

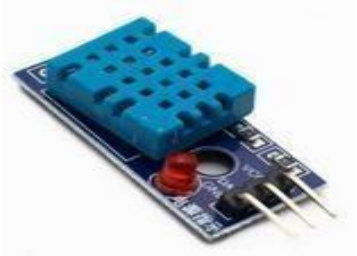

**Figure 2.4: DHT11**

### 2.2.5 UltraSonic Distance Sensor

The Ultrasonic Distance Sensor is a widely used module for measuring distances by emitting ultrasonic waves and calculating the time taken for the echo to return. This measurement can be used for various applications, including object detection and level monitoring. The output of this sensor is an analog or digital reading, depending on the model.

Ultrasonic sensors can be affected by environmental factors such as temperature and humidity, which may introduce noise into the readings. To minimize this interference, advanced signal processing techniques can be applied. The working principle of the ultrasonic sensor is similar to sonar technology, where sound waves reflect off surfaces and return to the sensor, allowing accurate distance calculations.

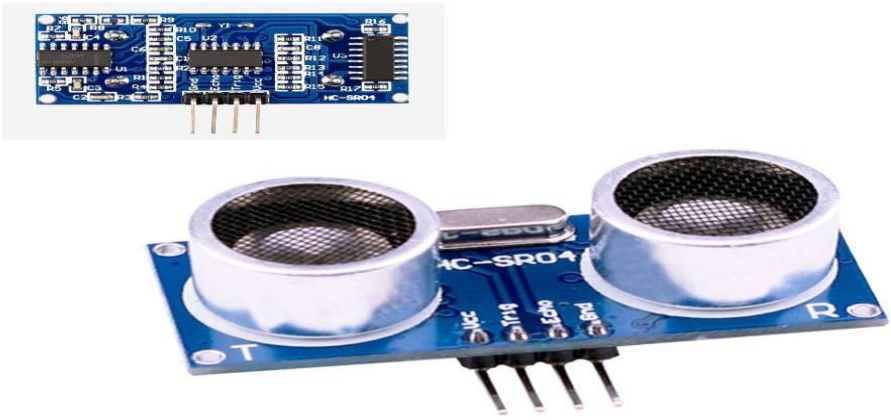


**Figure 2.5: UltraSonic Sensor**

# Chapter 3

# Problem Description

# CHAPTER-3
# PROBLEM DESCRIPTION

In modern urban management, digital solutions have become increasingly essential, especially for efficient waste disposal. Overflowing garbage bins lead to unsanitary conditions, environmental hazards, and increased mosquito breeding. To address these challenges, the IoT-Based Smart Bin is developed using an ESP32 microcontroller, Ultrasonic sensor, PIR sensor, DHT11 sensor, and the ThingSpeak cloud platform. This system enables real-time monitoring of waste levels, temperature, and human activity, allowing for better waste management and timely interventions.

The system operates by collecting data from various sensors and transmitting it to the ThingSpeak platform, where it is visualized in real-time. Ultrasonic sensors measure the garbage level, PIR sensors detect human presence near the bin, and DHT11 sensors monitor temperature and humidity. The ESP32 microcontroller, with built-in Wi-Fi capabilities, ensures seamless communication between the sensors and the cloud. Red, yellow, and green indicator lights provide a simple visual cue for passersby to assess the bin's status and notify municipal authorities when necessary.

One of the main advantages of this system is its cost-effectiveness. The components used—such as the ESP32 and ultrasonic sensors—are affordable, making it a practical and scalable solution for cities and municipalities. By ensuring real-time waste level monitoring, this system helps prevent overflowing bins, reduces manual inspections, and improves cleanliness, especially during festivals and high-waste periods.

By leveraging ThingSpeak, waste management teams can easily access and analyze data trends via web or mobile interfaces, making the system user-friendly and efficient. This data-driven approach enables authorities to prioritize garbage collection, reduce unnecessary waste pickups, and optimize resource allocation. The system also helps in minimizing mosquito breeding and maintaining a hygienic urban environment.

The future scope of this system includes integrating AI and machine learning algorithms to predict garbage accumulation trends and optimize collection schedules. Additional sensors, such as gas sensors, could be incorporated to detect hazardous gases in waste bins, further enhancing urban cleanliness and safety.

Overall, the IoT-Based Smart Bin provides a smart, scalable, and cost-effective solution for real-time waste management. By utilizing IoT technology, this system enhances urban hygiene, promotes smarter cities, and ensures a more efficient, sustainable, and environmentally friendly waste disposal process.

# Chapter 4
# Literature Survey

# CHAPTER-4
# LITERATURE SURVEY

The Internet of Things (IoT) is a transformative technology that connects physical devices to the internet, enabling them to collect, share, and analyze data in real-time. In the context of smart cities, IoT plays a crucial role in optimizing urban infrastructure and services, such as transportation, energy, and waste management. By integrating IoT into waste management systems, cities can monitor garbage levels, track collection schedules, and allocate resources more efficiently. This not only reduces operational costs but also improves public health and cleanliness. IoT-based solutions are scalable and adaptable, making them suitable for diverse urban environments. For example, smart bins equipped with IoT sensors can provide real-time data to municipal authorities, enabling them to prioritize waste collection and prevent overflow incidents. According to **Zanella et al. (2014)**, IoT technologies are essential for creating sustainable and efficient smart cities, with waste management being a key application area.[1]

Smart waste management systems leverage advanced technologies like sensors, microcontrollers, and cloud platforms to automate and optimize waste collection processes. In your project, the smart bin uses an ultrasonic sensor to measure the level of garbage inside the bin. This data is transmitted to a cloud platform, where it is visualized in real-time, allowing municipal authorities to monitor bin status remotely. Additionally, a Passive Infrared (PIR) sensor is used to detect human presence near the bin, which can encourage public participation in waste management. For instance, if a passerby notices a red light (indicating a high garbage level), they can alert the municipal corporation for timely cleanup. These systems not only improve efficiency but also reduce the environmental impact of waste accumulation. **Anagnostopoulos et al. (2017)** highlight the challenges and opportunities of IoT-enabled waste management systems, emphasizing their role in improving urban cleanliness and resource allocation.[2]

The smart bin incorporates three key sensors: an ultrasonic sensor, a PIR sensor, and a DHT11 sensor. The ultrasonic sensor measures the distance between the sensor and the garbage surface, providing accurate data on the bin's fill level. This information is displayed using colored lights (red for high, yellow for medium, and green for low levels), which help passersby and municipal workers assess the bin's status at a glance. The PIR sensor detects motion near the bin, ensuring that the system is aware of human presence. This feature can be used to trigger alerts or notifications for maintenance. The DHT11 sensor monitors temperature and humidity inside the bin, which is crucial for preventing issues like odor, decomposition, and mosquito breeding. Together, these sensors

create a comprehensive monitoring system for efficient waste management. **Kumar et al. (2016)** discuss the use of ultrasonic sensors and microcontrollers in IoT-based garbage alert systems, demonstrating their effectiveness in real-time monitoring.[3]

Cloud platforms like ThingSpeak are essential for IoT-based projects as they enable real-time data visualization and analysis. In your project, the ESP32 microcontroller collects data from the ultrasonic, PIR, and DHT11 sensors and transmits it to ThingSpeak. The platform generates graphs and dashboards that display garbage levels, temperature, humidity, and human presence over time. Municipal authorities can access this data remotely and use it to prioritize waste collection routes, especially during high-waste periods like festivals. The cloud platform also stores historical data, which can be analyzed to identify trends and improve long-term waste management strategies. By leveraging cloud technology, your smart bin system ensures transparency, efficiency, and scalability. **MathWorks' ThingSpeak documentation** provides detailed insights into the platform's capabilities for IoT data visualization and analysis.

One of the key advantages of my smart bin system is its contribution to public health and environmental sustainability. Improper waste management can lead to the accumulation of garbage, which attracts mosquitoes and other disease-carrying pests. By monitoring garbage levels and ensuring timely collection, your system helps reduce mosquito breeding and the spread of vector-borne diseases like dengue and malaria. Additionally, the DHT11 sensor's temperature and humidity data can be used to identify conditions that promote waste decomposition and odor, enabling proactive measures to maintain hygiene. During festive seasons, when waste generation peaks, the smart bin system ensures that bins are emptied promptly, preventing overflow and maintaining cleanliness in public spaces. **Kulkarni and Sutar (2019)** highlight the public health benefits of IoT-based waste management systems, particularly in reducing mosquito breeding and improving urban hygiene.[5]

The ESP32 microcontroller serves as the brain of your smart bin system, integrating data from all sensors and transmitting it to the cloud platform. Known for its low power consumption and Wi-Fi capabilities, the ESP32 is ideal for IoT applications. It processes data from the ultrasonic sensor to determine garbage levels, reads temperature and humidity values from the DHT11 sensor, and detects human presence using the PIR sensor. The microcontroller then sends this data to ThingSpeak via Wi-Fi, where it is visualized and analyzed. The ESP32's versatility and affordability make it a popular choice for IoT projects, ensuring seamless integration of hardware and software

components in your smart bin system. **Ramson and Moni (2017)** discuss the use of microcontrollers like the ESP32 in IoT applications, emphasizing their role in enabling real-time data processing and communication.

ThingSpeak is a cloud-based IoT platform that allows users to collect, analyze, and visualize sensor data in real-time. In your project, ThingSpeak receives data from the ESP32 microcontroller and generates graphs for garbage levels, temperature, humidity, and human presence. These visualizations help municipal authorities monitor bin status remotely and make data-driven decisions. For example, if the garbage level graph shows a consistent increase in a particular area, authorities can allocate more resources to that location. ThingSpeak also supports alerts and notifications, which can be configured to notify maintenance teams when a bin reaches its capacity. By providing actionable insights, ThingSpeak enhances the efficiency and effectiveness of your smart bin system. **Patil and Kale (2016)** discuss the integration of IoT and cloud platforms for waste management, highlighting their role in enabling remote monitoring and decision-making.

# Chapter 5

# Software Requirement Specifications

# CHAPTER 5
# SOFTWARE REQUIREMENTS SPECIFICATION

Software Requirements Specification (SRS) for IoT based Smart Bin using ESP32, DHT11, PIR Sensor, UltraSonic Sensor.

## 5.1 Functional Requirements

Functional requirements define the behavior of the software system. They describe what the system should do and how it should behave under different circumstances. The functional requirements for the IoT based Smart Bin using ESP32, DHT11, PIR Sensor, UltraSonic Distance Sensor are as follows:

1. Detect Garbage Levels using an Ultrasonic Sensor and classify the bin status as High (Red), Medium (Yellow), or Low (Green).
2. Detect Human Presence near the bin using a PIR Sensor to monitor illegal dumping or vandalism.
3. Monitor Temperature and Humidity inside the bin using a DHT11 Sensor to help prevent odor and mosquito breeding.
4. Transmit Real-Time Data to the ThingSpeak Cloud Platform for analysis and remote monitoring by municipal authorities.

## 5.2 Non-Functional Requirements

Non-functional requirements define the characteristics of the IoT-based Smart Bin system. They describe how the system should perform in terms of speed, reliability, security, and usability. The non-functional requirements for the IoT-based Smart Bin using ESP32, DHT11, PIR Sensor, and Ultrasonic Sensor are as follows:

**Performance:**

The system should accurately detect garbage levels, human presence, temperature, and humidity. Data must be sent to the ThingSpeak cloud in real-time with minimal latency for efficient monitoring by municipal authorities.

**Reliability:**

The system must ensure consistent and accurate garbage level detection, with minimal chances of false readings. The traffic light indicators (Red, Yellow, Green) should update in real-time to assist passersby in notifying the municipal corporation.

**Security:**

Communication between the ESP32 microcontroller and ThingSpeak must be secure. The system should use encrypted protocols to prevent unauthorized access or tampering with the data.

**Usability:**

The system should have a user-friendly design, ensuring that passersby can easily interpret the light indicators and notify the authorities. The ThingSpeak dashboard should present real-time graphs of garbage levels, temperature, humidity, and human detection for better decision-making.

**Compatibility:**

The system should be compatible with various Wi-Fi networks and should be scalable to integrate additional sensors like gas sensors for air quality monitoring. It should also support future enhancements, such as AI-based analytics for waste collection optimization.

Overall, the IoT-based Smart Bin system ensures efficient waste management, hygiene improvement, and mosquito control by leveraging real-time IoT-based monitoring and smart alerts.

# Chapter 6

# Software & Hardware Design

# CHAPTER 6
# HARDWARE & SOFTWARE DESIGN

## 6.1 Hardware Design

The architecture for the IoT based Smart Bin using ESP32, DHT11, PIR Sensor, UltraSonic Sensor consists of the following components:

### 6.1.1 UltraSonic Distance sensors

The Ultrasonic Sensor is used to measure the garbage level and transfers the data to the cloud platform using the ESP32 Wi-Fi module.

### 6.1.2 ESP32

The ESP32 development board is used to control the Ultrasonic, PIR, and DHT11 sensors and send the status of the Smart Bin to a cloud-based server using Wi-Fi connectivity. It receives data from these sensors and transmits it to the ThingSpeak platform for real-time monitoring.

### 6.1.3 Cloud-based Server (ThingSpeak)

The cloud-based server, ThingSpeak, is used to store the Smart Bin's data, including garbage level, temperature, humidity, and PIR sensor readings. It provides real-time monitoring capabilities, allowing municipal authorities to remotely access bin status through a web or mobile interface, enabling efficient waste management and timely cleanliness interventions.

### 6.1.4 Web or Mobile Application

The web or mobile application is used to monitor the Smart Bin's status in real-time. It retrieves data from the ThingSpeak cloud server and presents it in a user-friendly interface. Notifications or alerts can also be sent to municipal authorities via messaging applications like Telegram in case of high garbage levels or abnormal environmental conditions.

The architecture of the IoT based Smart Bin is shown in the following diagram:

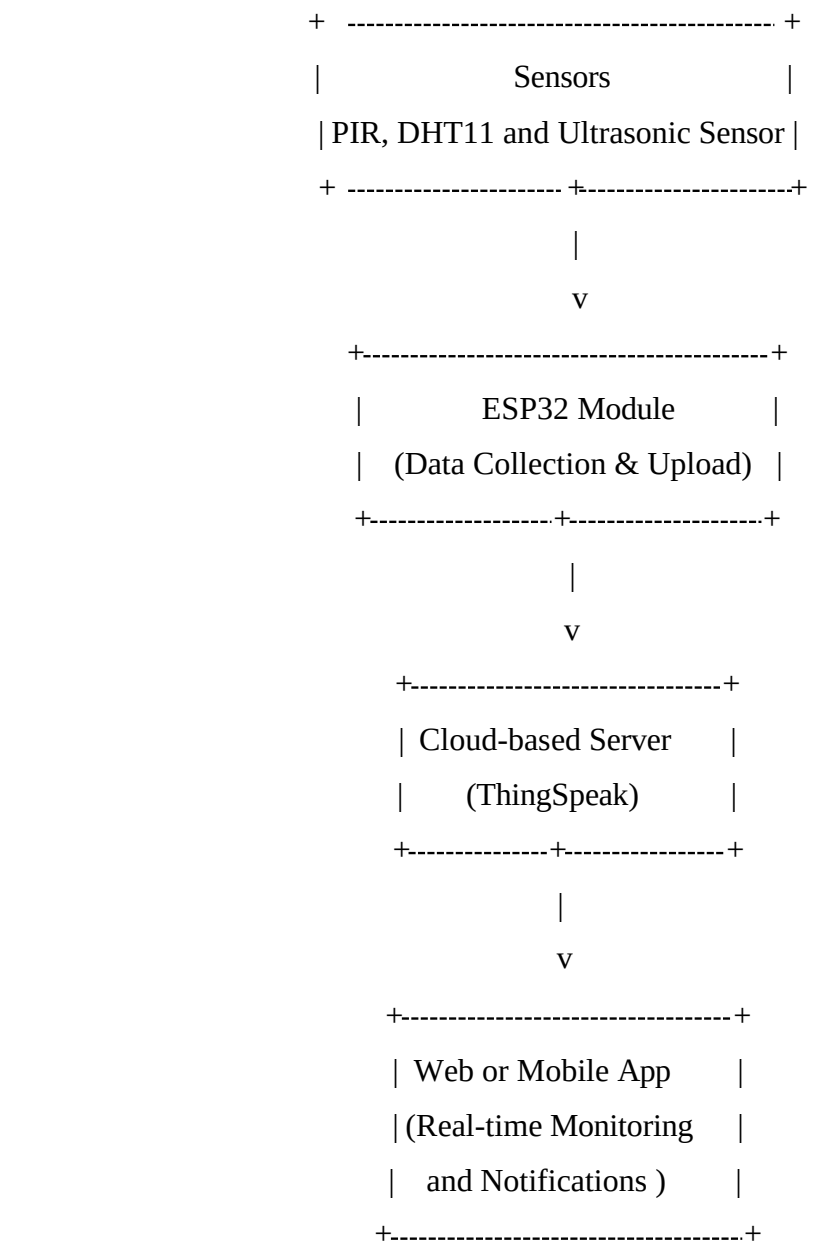


**Figure 6.1:Architecture Diagram**

The sensors (Ultrasonic, PIR, and DHT11) are connected to the ESP32 microcontroller, which collects data on garbage levels, human presence, temperature, and humidity. The ESP32 sends this data to the ThingSpeak cloud server using Wi-Fi connectivity. The ThingSpeak cloud server stores the waste management data in a database and provides real-time monitoring capabilities. A web or mobile application fetches data from the cloud-based server and displays it in an intuitive interface. Additionally, alerts and notifications are sent to municipal authorities via Telegram when the garbage level is high or environmental conditions require immediate attention.

### 6.1.4 Circuit Diagram

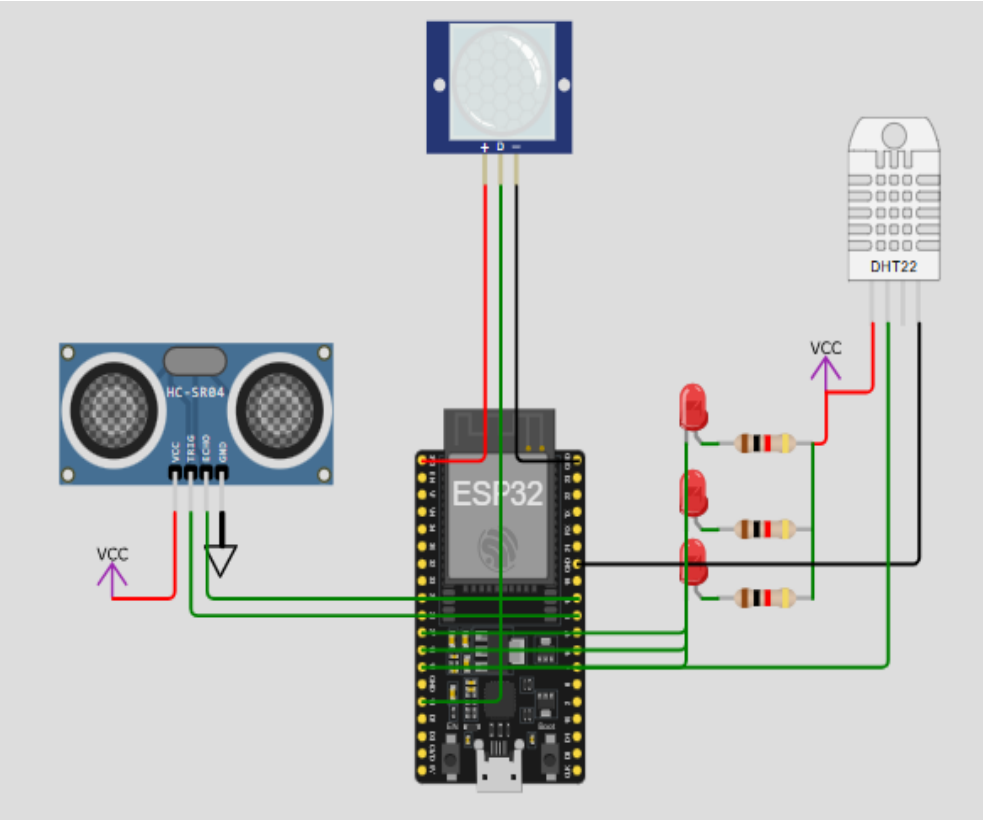


**Figure 6.2: Circuit Diagram**

### 6.1.5 Pin Diagram

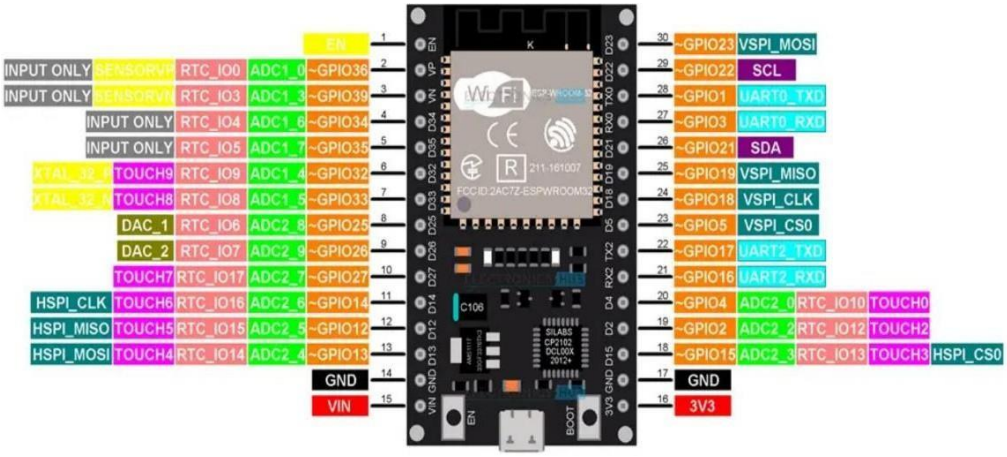


**Figure 6.3: Pin Diagram**

## 6.2 Software Design

The IoT-Based Smart Bin uses ESP32 to collect data from Ultrasonic, PIR, and DHT11 sensors, displaying garbage levels through indicator lights, sending data to ThingSpeak via Wi-Fi, and alerting municipal authorities via Telegram.

### 6.2.1 Flowchart

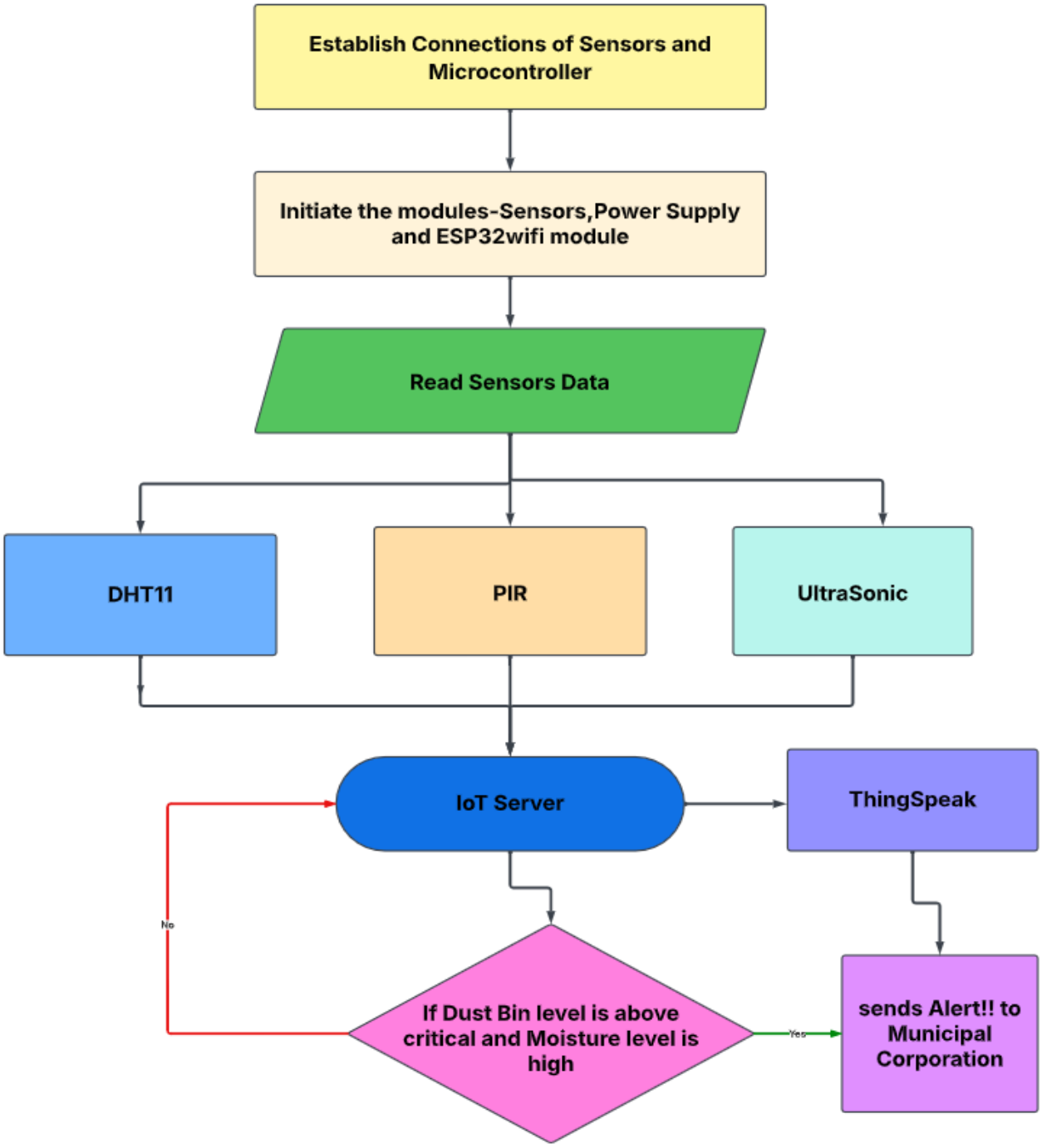


**Figure 6.4: Flowchart**

### 6.2.2 Data Flow Diagram

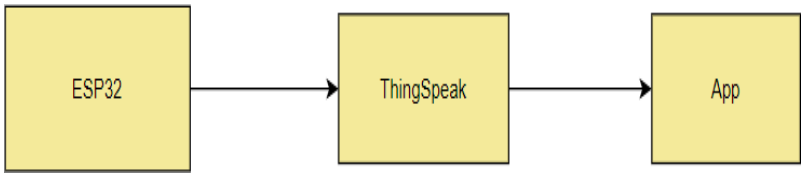


**Figure 6.5: DFD Diagram**

### 6.2.3 Use Case Diagram

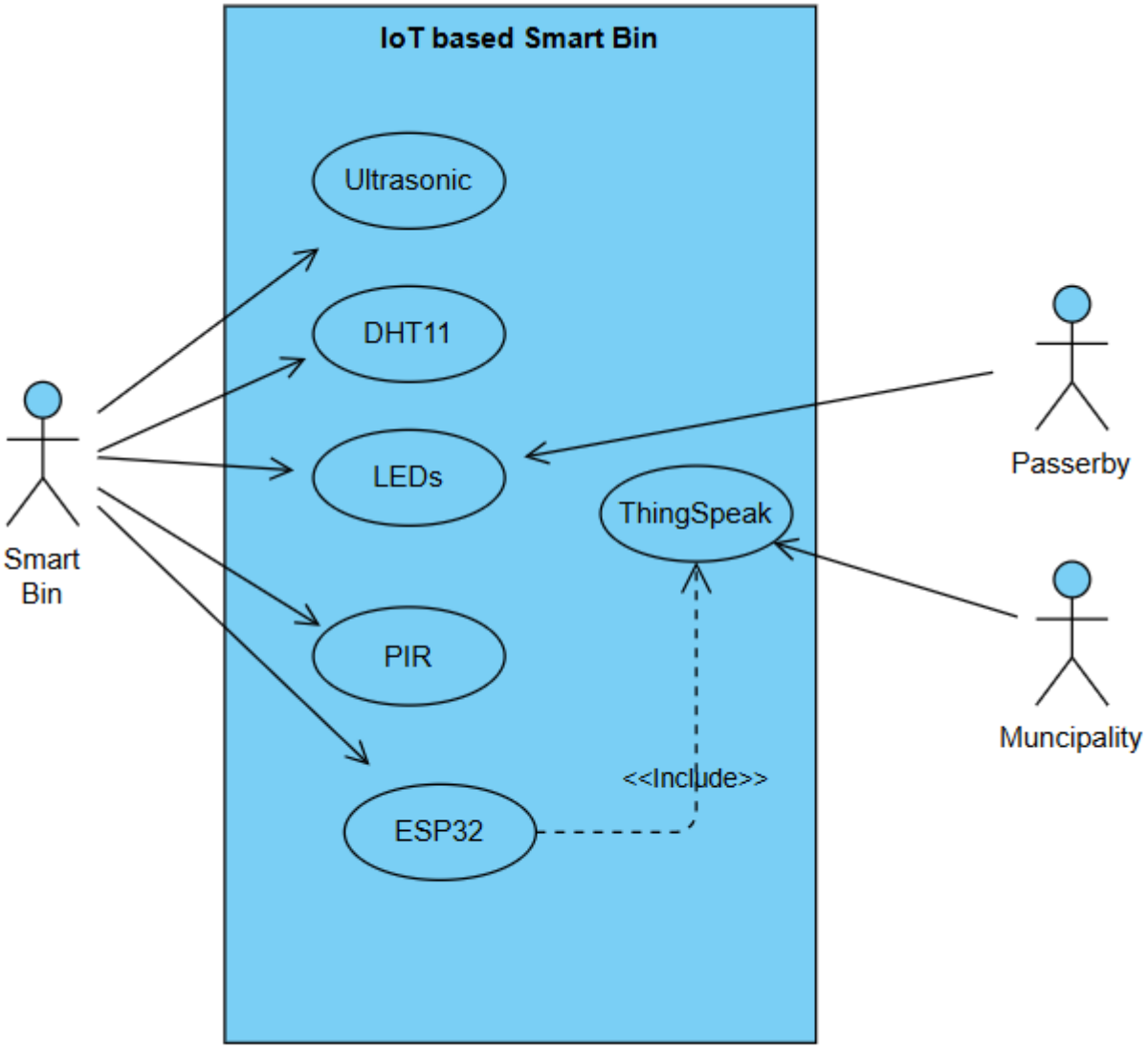


**Figure 6.6: Use Case Diagram**

# Chapter 7
# IoT Module

# CHAPTER 7
# IoT MODULE

## 7.1 Pre-Processing Steps for ESP32 and Sensors in IoT-Based Smart Bin

In an IoT-based Smart Bin using the ESP32, pre-processing is essential to ensure that the raw data from the sensors (Ultrasonic, PIR, and DHT11) is clean, accurate, and ready for analysis or transmission to the cloud server. Sensor readings may contain noise, errors, or irrelevant fluctuations that must be handled before storage or analysis.

**Steps Involved in Pre-Processing:**

### 7.1.1. Data Filtering:

Sensor readings from the Ultrasonic, PIR, and DHT11 sensors may contain noise due to environmental factors, electromagnetic interference, or incorrect sensor placements. Filtering techniques such as smoothing functions and outlier removal can help refine data accuracy. For example:

- **Ultrasonic Sensor:** Filtering out erroneous spikes in distance readings.
- **PIR Sensor:** Reducing false motion detection caused by sudden temperature changes.
- **DHT11 Sensor:** Eliminating abrupt temperature and humidity variations caused by sensor latency.

### 7.1.2. Data Transformation:

Raw data from sensors may require scaling, normalization, and conversion to appropriate units to ensure uniformity.

- **Ultrasonic Sensor:** Converting distance measurements to predefined garbage level categories (low, medium, high).
- **DHT11 Sensor:** Standardizing temperature and humidity values for accurate trend analysis.
- **Timestamp Alignment:** Synchronizing data points with consistent timestamps (e.g., UTC format) for real-time monitoring

### 7.1.3. Data Validation:

Validation checks ensure that the data received is complete and accurate.

- **Ultrasonic Sensor:** Ensuring correct distance readings to avoid false bin level status.
- **PIR Sensor:** Validating motion detection only when movement occurs near the bin.
- **DHT11 Sensor:** Checking that temperature and humidity readings fall within expected environmental limits.

### 7.1.4 Data Segmentation:

For effective waste management, sensor data is segmented at regular intervals (e.g., every 5 minutes) to analyze garbage accumulation and environmental conditions. The Ultrasonic Sensor monitors bin levels for optimizing waste collection, the DHT11 Sensor tracks temperature-related decomposition effects, and the PIR Sensor detects motion trends to assess pedestrian interactions with the bin.

**Detailed Pre-Processing Steps:**

- **Eliminating Duplicates:** Ensuring unique sensor readings to prevent redundant data storage.
- **Correcting Inconsistent Data:** Applying interpolation techniques to estimate missing or inconsistent values in sensor readings.
- **Signal Smoothing:** Using smoothing algorithms (e.g., moving average filters) to reduce fluctuations in sensor data for better accuracy.

## 7.2 IoT Components

### 7.2.1 Software (IDE)

The Arduino IDE is an open-source software, which is used to write and upload code to the Arduino boards. The IDE application is suitable for different operating systems such as Windows, Mac OS X, and Linux. It supports the programming languages C and C++. Here, IDE stands for Integrated Development Environment.

The program or code written in the Arduino IDE is often called as sketching. We need to connect the Genuino and ESP32 board with the IDE to upload the sketch written in the Arduino IDE software. The sketch is saved with the extension '.ino'

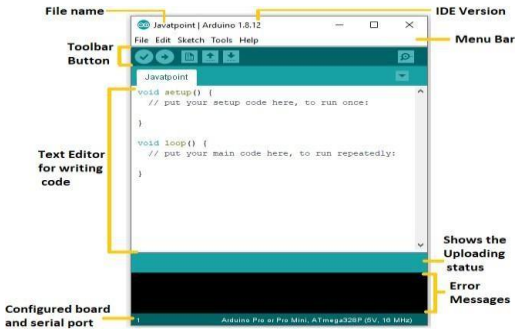


**Figure. 7.1: IDE**

### 7.2.2 ESP 32

- **Overview :**ESP32 is a low-cost, low-power system on a chip (SoC) developed by Espressif Systems, widely used in IoT applications for its versatility and robust wireless communication features.
- **Microcontroller :** It is based on a dual-core Tensilica Xtensa LX6 microprocessor (or single-core in some models), running at up to 240 MHz
- **Wi-Fi Connectivity :** The ESP32 includes built-in Wi-Fi (802.11 b/g/n) capabilities, allowing it to connect to wireless networks and function as both a client and an access point.
- **Bluetooth :** It supports both Bluetooth 4.2 and Bluetooth Low Energy (BLE), making it ideal for short-range wireless communication in IoT applications.
- **Low Power Consumption :** ESP32 is designed for power efficiency, featuring multiple low-power modes (deep sleep, light sleep), which makes it ideal for battery-operated IoT devices.
- **Cost-Effective :** The ESP32 is highly affordable, making it a popular choice for DIY projects, prototyping, and mass production in IoT solutions.
- **Real-Time Clock (RTC) :** The chip has an integrated RTC that allows timekeeping and low-power functionality while the device is in deep sleep mode.
- **Dual Core Processing :** It can run two processes simultaneously, offering better performance for more complex applications.
- **Wide Temperature Range :** The ESP32 can operate in temperatures ranging from -40°C to 125°C, making it suitable for industrial and outdoor applications.


**Figure 7.2: ESP 32**

### 7.2.3 UltraSonic Distance Sensor

- **Garbage Level Monitoring:** The Ultrasonic Distance Sensor measures the waste level inside the bin, helping determine whether it is full, partially filled, or empty.

- **Real-Time Data Collection:** It continuously tracks garbage accumulation and updates the data on the ThingSpeak cloud platform.

- **Optimized Waste Collection:** By detecting fill levels, it allows municipal corporations to prioritize waste collection schedules, improving efficiency.

- **Non-Contact Measurement:** Uses ultrasonic waves for distance measurement, ensuring accurate readings without physical contact with the waste.

- **Applications:** Widely used in IoT-based smart waste management systems, automated garbage bins, and urban sanitation monitoring.


**Figure 7.3: UltraSonic sensor**

**Figure 7.4: Waste Segregation**

**Table 7.1: Dengue cases in Delhi**

| Age (years) | IgM ELISA-positive dengue cases (%) |
|---|---|
| 0-10 | 248 (14.6) |
| 11-20 | 393 (23.1) |
| 21-30 | 524 (30.8) |
| 31-40 | 273 (16.1) |
| 41-50 | 131 (7.7) |
| >50 | 131 (7.7) |
| Total | 1700 (100) |

ELISA: Enzyme-linked immunosorbent assay

### 7.2.4 DHT11

- **Temperature and Humidity Measurement:** The DHT11 sensor measures temperature (0–50°C) and relative humidity (20–90%) with reasonable accuracy, making it suitable for basic environmental monitoring.

- **Digital Output:** It provides data in digital form, eliminating the need for complex analog-to-digital conversion.

- **Compact and Affordable:** The sensor is small, low-cost, and widely available, making it ideal for budget-friendly IoT projects.

- **Low Power Consumption:** It is energy-efficient, requiring minimal power, which makes it suitable for battery-operated devices.

- **Ease of Integration:** The DHT11 is easy to interface with microcontrollers like ESP32 using a single data pin, making it beginner-friendly for DIY projects and prototyping.

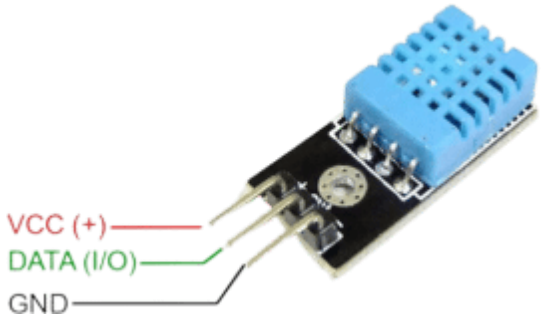


**Figure 7.5: DHT11**

### 7.2.5 Bread Board

- **Prototyping Made Easy:** A breadboard allows for quick and temporary construction of electronic circuits without the need for soldering, making it ideal for testing and experimentation.

- **Reusable and Versatile:** Components can be easily added, removed, or rearranged, enabling modifications to the circuit design without damaging the components or board.

- **Wide Compatibility:** Breadboards are compatible with a variety of components, such as resistors, capacitors, LEDs, and microcontrollers, making them useful for a wide range of projects.

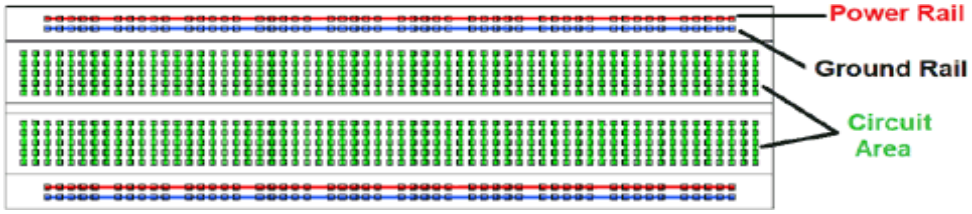


**Figure 7.6: Breadboard**

### 7.2.6 Jumper Wires

- **Easy Connections:** IoT jumper wires are essential for creating quick and secure connections between components like sensors, microcontrollers (e.g., ESP32), and breadboards without soldering.

- **Versatile Types:** They come in male-to-male, male-to-female, and female-to-female configurations, allowing flexibility in connecting different pin types and components.

- **Reusable and Durable:** Made with insulated wires and durable connectors, jumper wires can be reused multiple times, making them ideal for prototyping and IoT projects.


**Figure 7.7: Jumper Wires**

## 7.3 Data Visualization for ESP32 and Sensors in IoT-Based Smart Bin

Data visualization is crucial in an IoT-based smart bin system using ESP32 and various sensors. By visualizing real-time data on garbage levels, temperature, humidity, and motion detection, municipal corporations can optimize waste collection and enhance sanitation management.

**Features of Data Visualization:**

### 7.3.1. Real-Time Garbage Level Monitoring:

- A live, interactive dashboard displays the bin's fill level in real-time using data from the ultrasonic sensor.
- The system categorizes garbage levels with red (high), yellow (medium), and green (low) indicators to help passersby and authorities quickly assess waste accumulation.

### 7.3.2. Environmental Condition Tracking:

- The DHT11 sensor records temperature and humidity data, which is displayed on the dashboard to analyze how environmental conditions affect waste decomposition and hygiene.
- Graphs visualize temperature and humidity trends over time, aiding in planning sanitation efforts.

### 7.3.3. Alert System for Waste Collection:

- Alerts and notifications are triggered when the bin reaches high capacity (red level), prompting municipal authorities to prioritize cleaning.
- PIR sensor data visualized on the dashboard helps monitor human activity around the bin, detecting improper usage or potential vandalism.

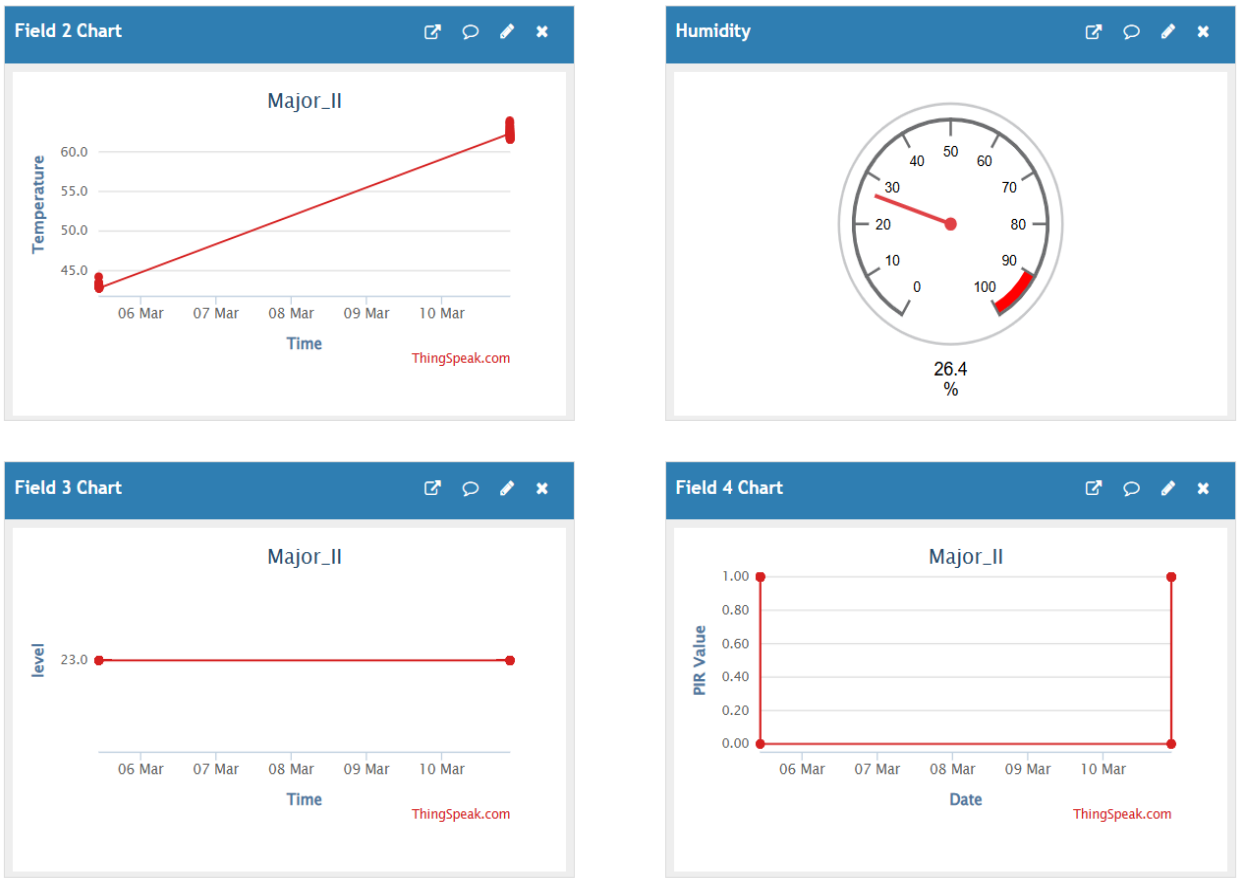


**Figure 7.8: Data Visualization**

# Chapter 8
# Coding

# CHAPTER 8
# CODING

## 8.1 Get Started with Arduino IDE

### 8.1.1 Installing the Arduino IDE

- Visit http://www.arduino.cc/en/main/software to download the latest Arduino IDE version for your computer's operating system. There are versions for Windows, Mac, and Linux systems. At the download page, click on the "Windows Installer" option for the easiest installation.
- Save the .exe file to your hard drive.
- Open the .exe file.
- Click the button to agree to the licensing agreement:


**Figure 8.1: Arduino Setup**

- Decide which components to install, then click "Next":
- Select which folder to install the program to, then click "Install":
- Wait for the program to finish installing, then click "Close":
- Now find the Arduino shortcut on your Desktop and click on it. The IDE will open up and you'll see the code editor:

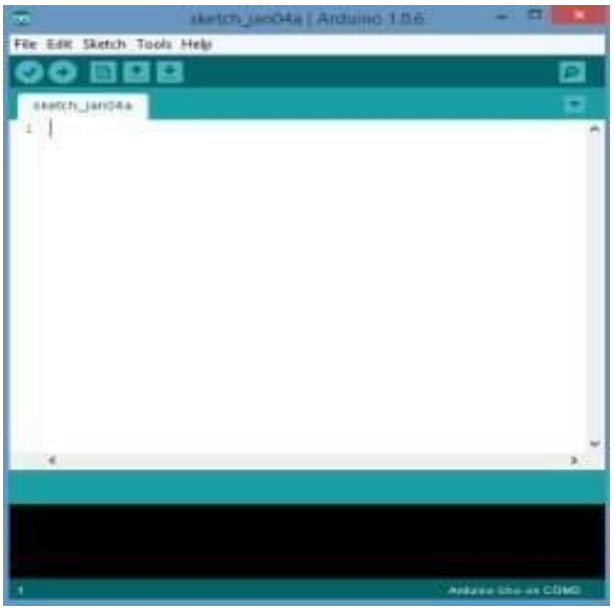


**Figure 8.2: Arduino IDE**

- Import important libraries according to board.
- Configuring the Arduino IDE according to the Board used like ESP 32.
- Choose correct PORT.
- Do necessity coding.

8_MODULE_BLURTOOTH_CONTROL.ino

```
#include <WiFi.h>
#include <HTTPClient.h>
#include "DHT.h"

// WiFi Credentials
const char* ssid = "SURYANSH.CSC 2.4G";
const char* password = "Divya@1985";

// ThingSpeak API Details
const char* server = "http://api.thingspeak.com/update";
const char* apiKey = "W739X7G8TT127GLZ";
const int channelID = 2430548;

// Pin Definitions
#define TRIG_PIN 5
#define ECHO_PIN 18
#define DHT_PIN 4
#define PIR_PIN 13
#define RED_LED 12
#define YELLOW_LED 14
#define GREEN_LED 27

```

```
// DHT11 Sensor
#define DHTTYPE DHT11
DHT dht(DHT_PIN, DHTTYPE);

void setup() {
    Serial.begin(9600); // Changed baud rate for better compatibility
    delay(1000);
    dht.begin();
    pinMode(TRIG_PIN, OUTPUT);
    pinMode(ECHO_PIN, INPUT);
    pinMode(PIR_PIN, INPUT_PULLDOWN); // Added internal pull-down
    pinMode(RED_LED, OUTPUT);
    pinMode(YELLOW_LED, OUTPUT);
    pinMode(GREEN_LED, OUTPUT);

    WiFi.begin(ssid, password);
    Serial.print("Connecting to WiFi");
    int attempt = 0;
    while (WiFi.status() != WL_CONNECTED && attempt < 20) { // 20 attempts (20 sec timeout)
        delay(1000);
        Serial.print(".");
        Serial.println(WiFi.status()); // Debugging WiFi status
        attempt++;
    }

    if (WiFi.status() == WL_CONNECTED) {
        Serial.println("\nConnected to WiFi!");
    } else {
        Serial.println("\nWiFi connection failed! Check SSID & Password.");
    }

    Serial.print("ThingSpeak Channel ID: ");
    Serial.println(channelID);
}

// Function to get distance from Ultrasonic Sensor
float getDistance() {
    digitalWrite(TRIG_PIN, LOW);
    delayMicroseconds(2);
    digitalWrite(TRIG_PIN, HIGH);
    delayMicroseconds(10);
    digitalWrite(TRIG_PIN, LOW);
    long duration = pulseIn(ECHO_PIN, HIGH);
    return (duration * 0.0343) / 2; // Convert to cm
}

void loop() {
    float temperature = dht.readTemperature();
    float humidity = dht.readHumidity();

    delay(100); // Allow PIR sensor to settle
    int pirValue = digitalRead(PIR_PIN);
    Serial.print("PIR Value: ");
    Serial.println(pirValue);

    float distance = getDistance();

```

```
    // Assuming bin height is 30 cm, calculate fill level
    int fillLevel = map(distance, 30, 0, 0, 100);
    fillLevel = constrain(fillLevel, 0, 100);

    // LED Indications
    digitalWrite(GREEN_LED, fillLevel <= 30);
    digitalWrite(YELLOW_LED, fillLevel > 30 && fillLevel <= 70);
    digitalWrite(RED_LED, fillLevel > 70);

    // Send data to ThingSpeak
    if (WiFi.status() == WL_CONNECTED) {
        HTTPClient http;
        http.begin(server);
        http.addHeader("Content-Type", "application/x-www-form-urlencoded");

        String postData = "api_key=" + String(apiKey) + "&field1=" + String(temperature) +
                          "&field2=" + String(humidity) + "&field3=" + String(fillLevel) +
                          "&field4=" + String(pirValue);

        int httpResponseCode = http.POST(postData);
        if (httpResponseCode > 0) {
            Serial.println("Data sent successfully to ThingSpeak: " + String(httpResponseCode));
        } else {
            Serial.println("Error sending data: " + http.errorToString(httpResponseCode));
        }
        http.end();
    } else {
        Serial.println("WiFi Disconnected! Reconnecting...");
        WiFi.begin(ssid, password);
    }

    delay(15000); // Send data every 15 seconds
}
```

## 8.2 Successful Compilation

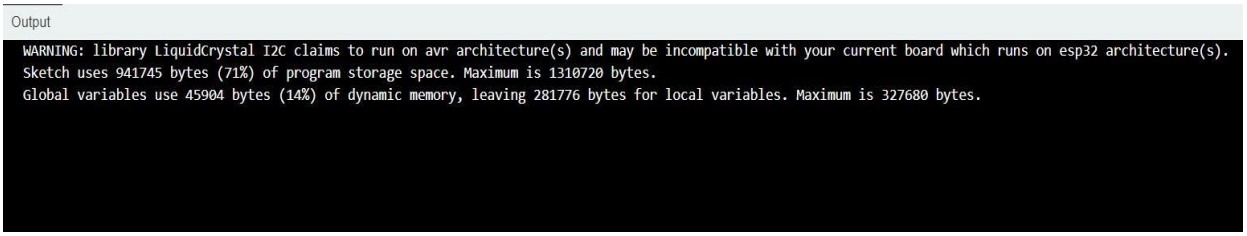


**Figure 8.3: Compilation**

## 8.3 Uploading code to Microcontroller

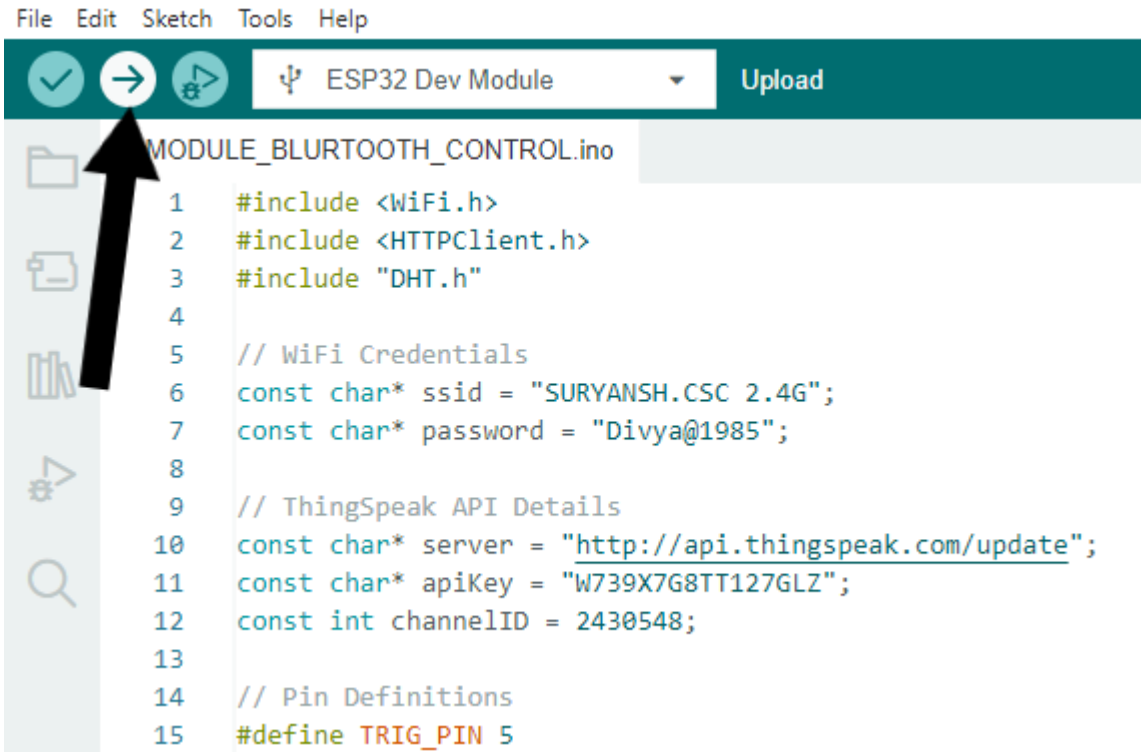


**Figure 8.4: Upload Button**

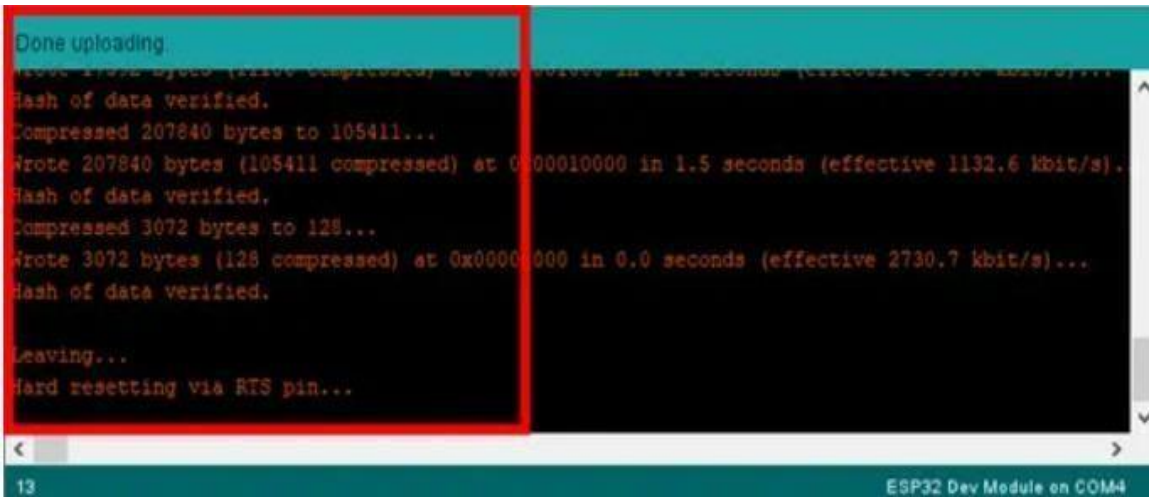


**Figure 8.5: Uploading**

# Chapter 9
# Result & Output Screens

# CHAPTER 9
# RESULTS & OUTPUT SCREENS

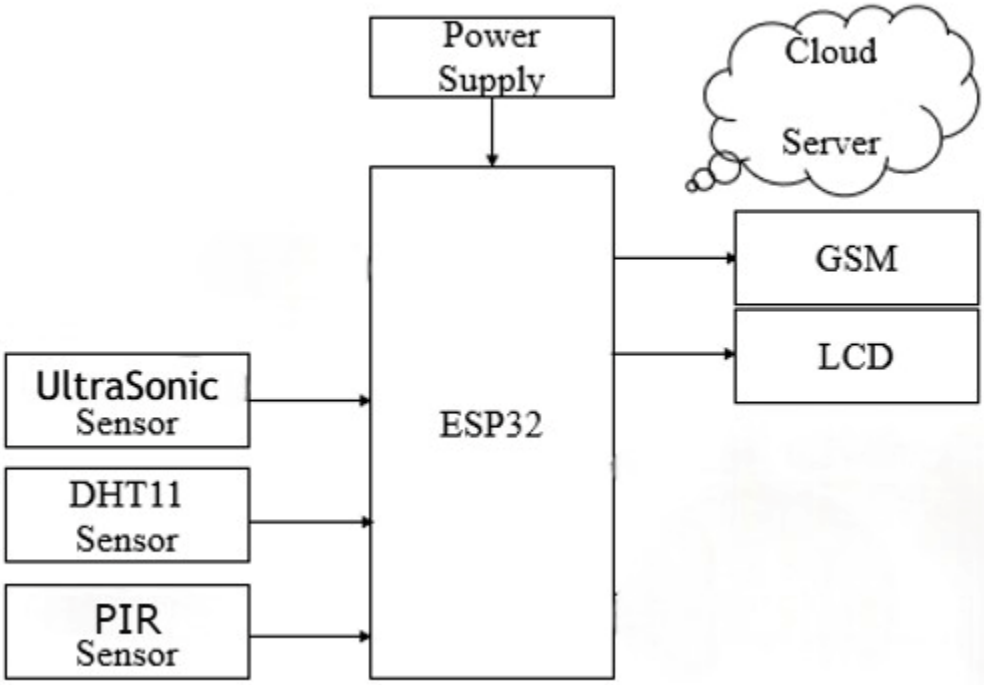


**Figure 9.1: Connections**

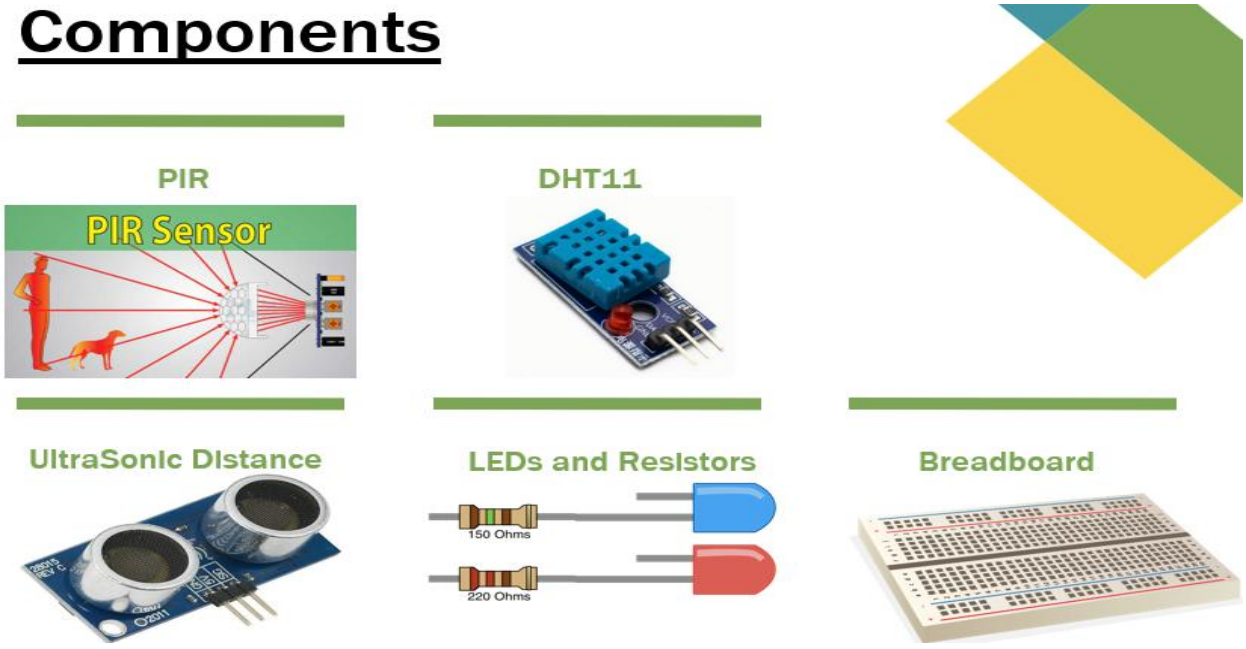


**Figure 9.2: Components**

**Figure 9.3: Bin working**

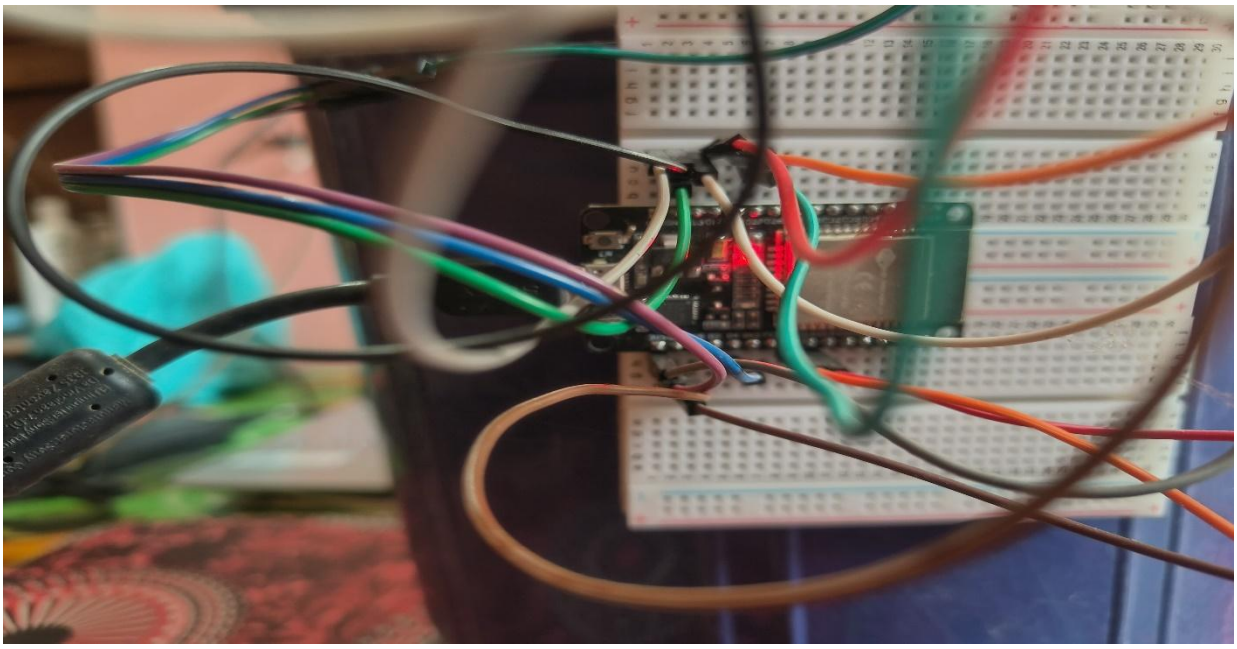

**Figure 9.4: Wire Connections Diagram**

**Figure 9.5: LEDs for passersby**

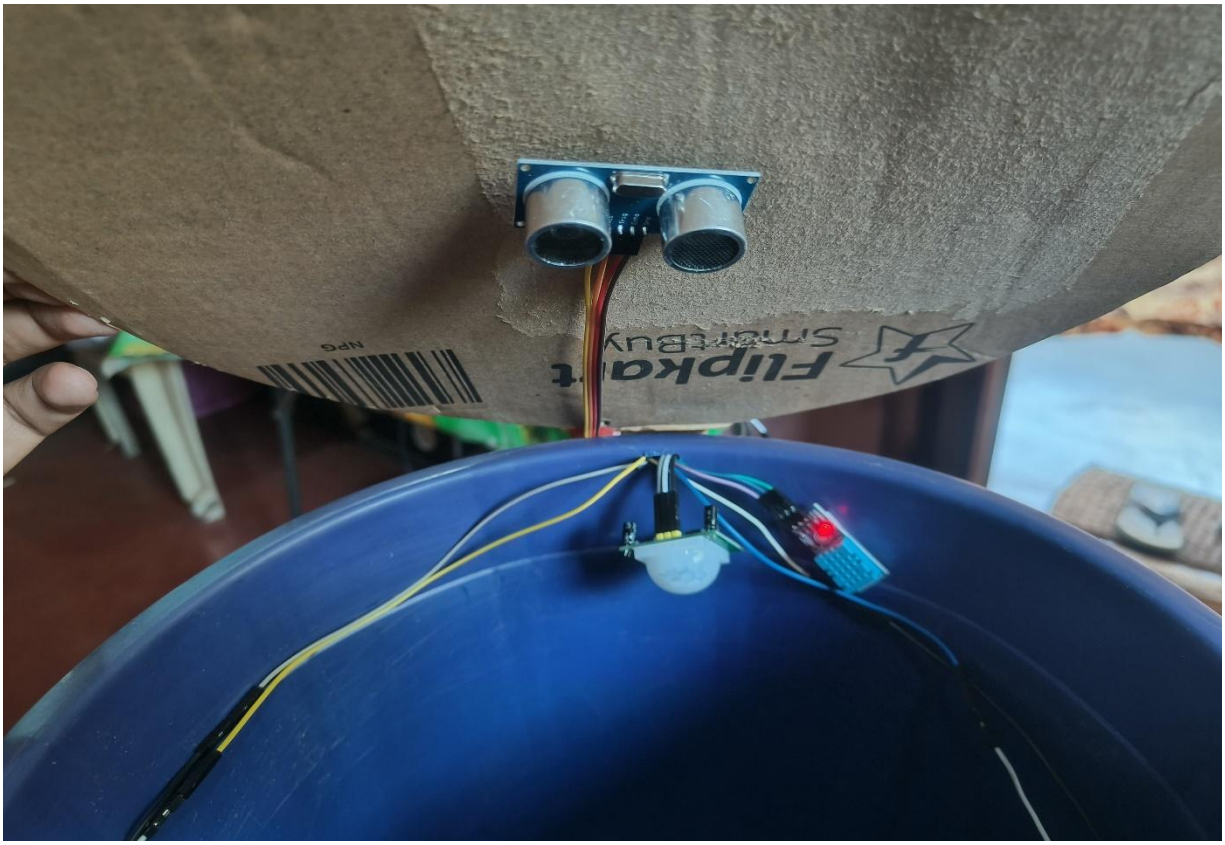

**Figure 9.6: Internal Sensors**

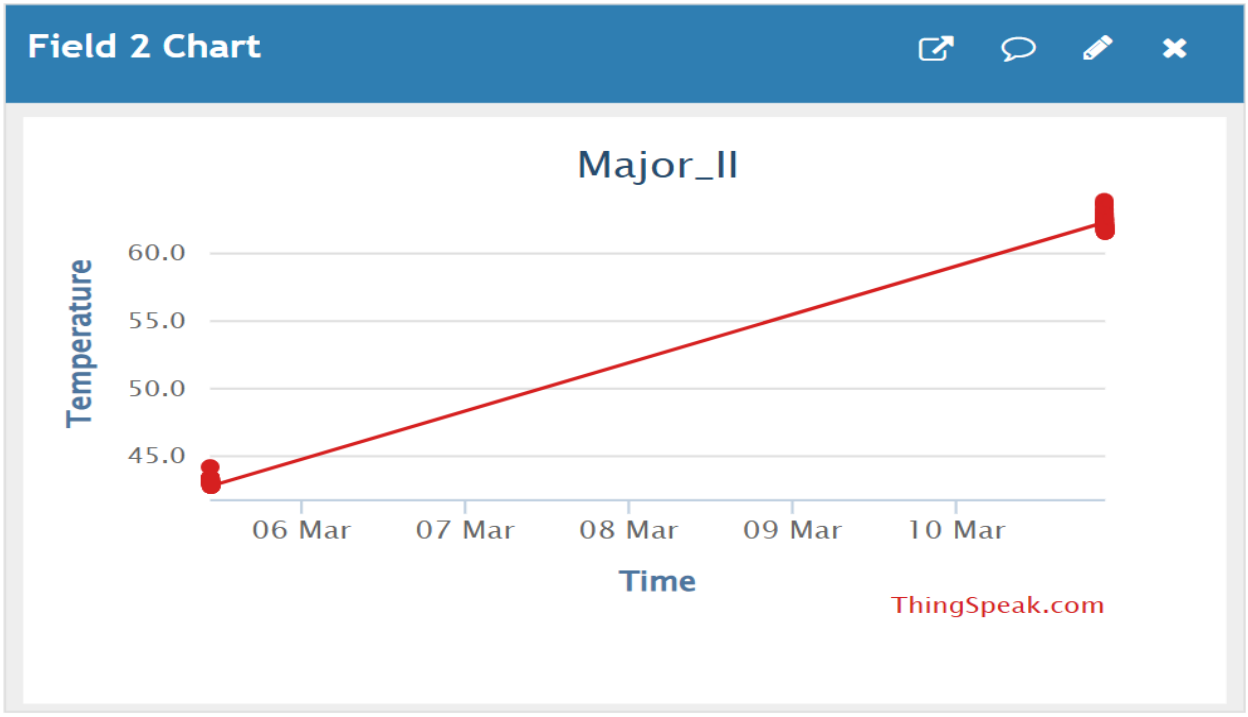


**Figure 9.7: Temperature Reading**

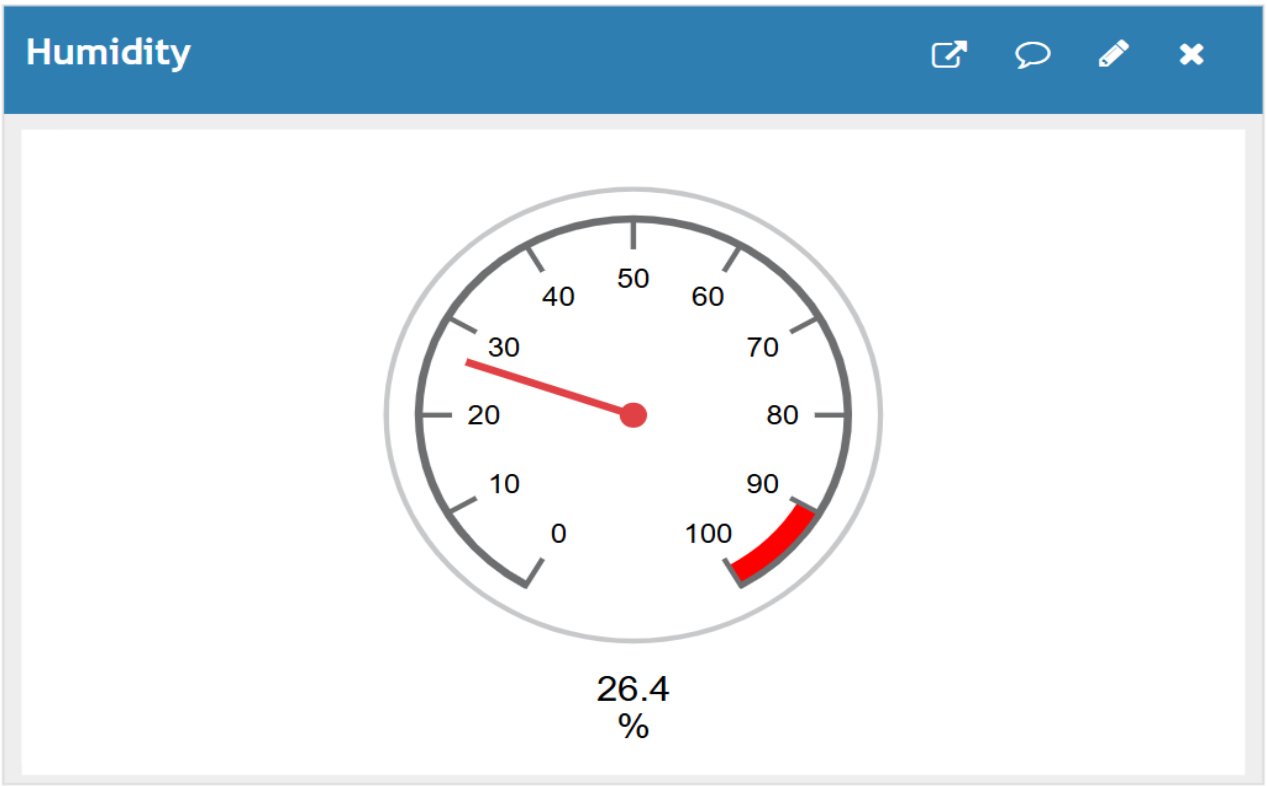


**Figure 9.8: Humidity**

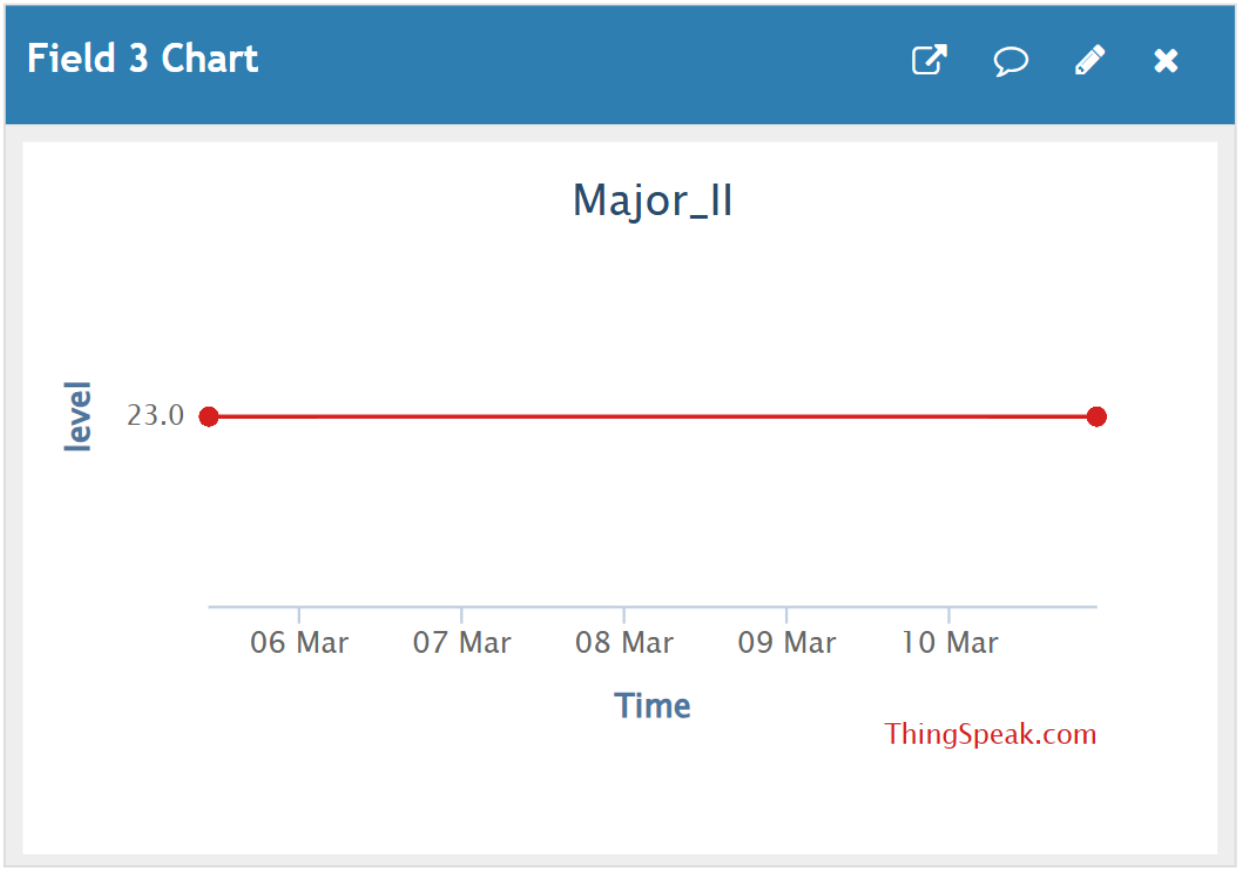


**Figure 9.9: Level Graph**

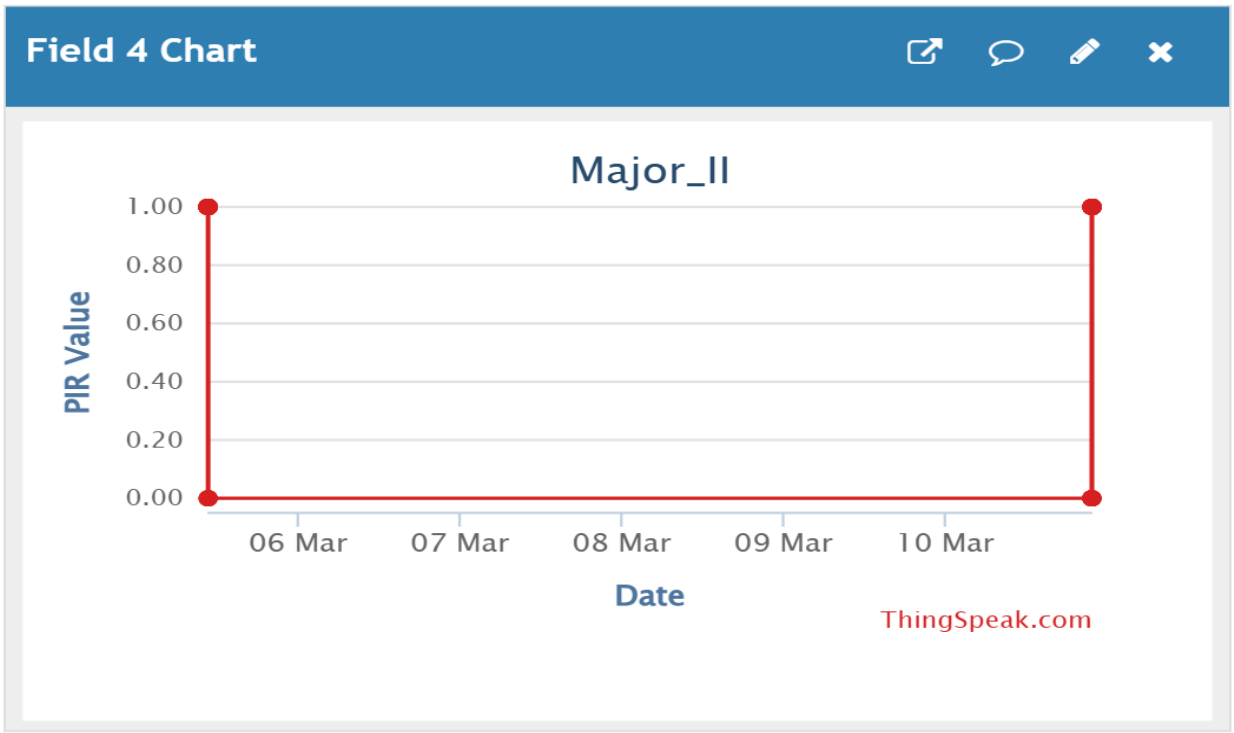


**Figure 9.10: PIR Graph**

# Chapter 10
# Conclusion & Future Work

# CHAPTER 10
# CONCLUSION & FUTURE WORK

## 10.1 CONCLUSION

The IoT-based Smart Bin is a significant innovation for Smart Cities, improving waste management through real-time monitoring and data analysis. Equipped with an ESP32 microcontroller, ultrasonic, PIR, and DHT11 sensors, it tracks garbage levels, detects human presence, and monitors environmental conditions, sending data to the ThingSpeak cloud. The red, yellow, and green indicator lights assist passersby in notifying municipal authorities for timely waste collection. This system helps optimize cleanliness, reduce mosquito breeding, and manage waste efficiently, particularly during festive seasons. By integrating IoT technology, the smart bin enhances urban sanitation, promotes sustainability, and supports a cleaner and healthier living environment, making it a valuable tool for smart waste management initiatives.

## 10.2 Future Work

- **Additional Sensors:** Integration of more sensors like an air vapor sensor for explosive detection could enhance safety and security in waste management.
- **Machine Learning:** AI-powered predictive analytics could help forecast waste accumulation trends, allowing municipal corporations to optimize garbage collection schedules.
- **Mobile App:** A dedicated mobile app would enable users and municipal authorities to monitor bin status, receive alerts, and analyze waste patterns conveniently.
- **Data Security:** Strengthened encryption and secure communication protocols will ensure that real-time waste management data remains protected from cyber threats.
- **AI Recommendations:** AI could provide insights on waste generation trends, helping authorities implement better waste disposal strategies and optimize resource allocation.
- **Automated Alerts:** Real-time notifications to municipal corporations when bins reach critical levels would improve efficiency and reduce overflow issues.

# REFERENCES

## JOURNALS & RESEARCH PAPERS


1. Zanella, A., Bui, N., Castellani, A., Vangelista, L., & Zorzi, M. (2014). **Internet of Things for Smart Cities**. *IEEE Internet of Things Journal*, 1(1), 22-32.
2. Anagnostopoulos, T., Zaslavsky, A., Kolomvatsos, K., Medvedev, A., Amirian, P., & Morley, J. (2017). **Challenges and Opportunities of Waste Management in IoT-Enabled Smart Cities: A Survey**. *IEEE Transactions on Sustainable Computing*, 2(3), 275-289.
3. Kumar, N. S., Vuayalakshmi, B., Prarthana, R. J., & Shankar, A. (2016). **IoT Based Smart Garbage Alert System Using Arduino UNO**. *IEEE Region 10 Conference (TENCON)*.
4. Patil, K. A., & Kale, N. R. (2016). **A Model for Smart Waste Management Using IoT and Cloud Computing**. *International Journal of Advanced Research in Computer and Communication Engineering*, 5(6), 234-239.
5. Kulkarni, S., & Sutar, M. (2019). **IoT-Based Smart Waste Management System for Smart Cities**. *International Journal of Scientific & Technology Research*, 8(11), 3146-3150.


## BOOKS


**6.** IEEE,' **IoT Based Smart Bin for Smart City Application**',Packt, Edition 1,2020.


## WEBSITES


7. https://www.researchgate.net/publication/323280336_IoT_Based_Waste_Management_for_Smart_City.
8. https://www.semanticscholar.org/paper/IoT-Based-Smart-Bin-for-Smart-City-Application-Mithinti-Kumar/f17d00ae9351ad0201b3dd66f5e5b32b85e887f5.

# PROJECT SUMMARY

## About Project

| | |
|---|---|
| **Title of the project** | IoT based Smart Bin. |
| **Semester** | 8th |
| **Members** | Abhay Prince, Amit Kumar, Amit Kumar Ankit Kishor |
| **Team Leader** | Abhay Prince |
| **Describe role of every member in the project** | Abhay Prince : IoT Circuit Connection<br>Amit Kumar : ThingSpeak integration<br>Ankit Kishor : Arduino Programming<br>Amit Kumar : Telegram integration |
| **What is the motivation for selecting this project?** | This project helps Municipality to prioritize the Smart Bin according to humidity and level of Garbage. |
| **Project Type**<br>**(Desktop Application, Web Application, Mobile App, Web)** | Web Application |

## Tools &Technologies

| | |
|---|---|
| **Programming language used** | C++ |
| **IDE used.**<br>**(With version)** | Arduino IDE |

| | |
|---|---|
| **Front End Technologies** **(With version, wherever Applicable)** | NA |
| **Back End Technologies** **(With version, wherever applicable)** | NA |
| **Database used.** **(With version)** | NA |

## Software Design & Coding

| | |
|---|---|
| **Is the prototype of the software developed?** | Yes |
| **SDLC model followed.** **(Waterfall, Agile, Spiral etc.)** | Agile |
| **Why above SDLC model is followed?** | Agile is a SDLC model that defines how software development needs to be done. It's not a single or specific method, and it is the collection of various methodologies and best practices that follow the value statement signed with the customer. |
| **Justify that the SDLC model mentioned above is followed in the project.** | Since, we didn't exactly know all the functionalities or the functionalities were frequently changing, we use Agile model, so that we could make desired changes whenever needed. |
| **Software Design approach followed.** **(Functional or Object Oriented)** | Object Oriented |

| | |
|---|---|
| **Name the diagrams. developed (According to the Design approach followed)** | Flowchart, Data Flow Diagram, Use Case Diagram |
| **In case Object Oriented The approach followed, by which of the OOPS principles are. covered in design?** | Inheritance |
| **No. of Tiers (Example 3-tier)** | NA |
| **Total no. of front-end pages** | 1 |
| **Total no. of tables in database** | NA |
| **Database is in which Normal Form?** | NA |
| **Are the entries in database encrypted?** | NA |
| **Front end validations applied (Yes / No)** | NA |
| **Session management done. (In case of web applications)** | NA |
| **Is application browser compatible? (In case of web applications)** | Yes |
| **Exception handling done. (Yes / No)** | Yes |
| **Commenting done in code (Yes / No)** | Yes |
| **Naming convention followed (Yes / No)** | Yes |
| | Cloud Integration and Displaying widgets on Web. |

| | |
|---|---|
| **What difficulties faced during deployment of the project?** | |
| **Total no. of Use-cases** | 1 |
| **Give titles of Use-cases** | IoT based Smart Bin. |

## *Project Requirements*

| | |
|---|---|
| **MVC architecture followed.** **(Yes / No)** | NA |
| **If yes, write the name of MVC architecture followed.** **(MVC-1, MVC-2)** | NA |
| **Design Pattern used.** **(Yes / No)** | No |
| **If yes, write the name of Design Pattern used** | |
| **Interface type** **(CLI / GUI)** | GUI |
| **No. of Actors** | 3 |
| **Name of Actors** | Passersby, Muncipality, Smart Bin |
| **Total no. of Functional Requirements** | 5 |
| **List a few important non-. Functional Requirements** | Performance, Reliability, Security, Usability. |

## ***Testing***

| | |
|---|---|
| **Which testing is performed?** **(Manual or Automation)** | Manual |
| **Is Beta testing done for this? project?** | No |

## ***Write project narrative covering above mentioned points***

The IoT-based Smart Bin leverages the ESP32 microcontroller to collect real-time waste management data from ultrasonic (garbage level), PIR (living body detection), and DHT11 (temperature and humidity) sensors. The system features three indicator lights—red for high level, yellow for medium level, and green for low level—allowing passersby to assess the bin's status and alert the municipal corporation when necessary.

All collected data is transmitted to the ThingSpeak cloud platform via Wi-Fi, where authorities can monitor waste levels, temperature, humidity, and PIR sensor activity through graphical visualizations. This enables the municipal corporation to prioritize bin cleanliness, optimizing waste collection schedules and reducing manual inspections.

This cost-effective system helps in reducing mosquito breeding and ensuring better hygiene, especially during festive seasons. Future enhancements include AI-driven waste level predictions, integration of gas sensors for detecting harmful gases, and mobile app support for improved monitoring, making urban waste management smarter and more efficient.


| | |
|---|---|
| **Abhay Prince** | **0187CS211003** |
| **Amit Kumar** | **0187CS211022** |
| **Amit Kumar** | **0187CS211023** |
| **Ankit Kishor** | **0187CS211029** |


**Guide Signature**

**Prof. Mayank Kurchaniya**

# APPENDIX 1

# GLOSSARY OF TERMS

## (In alphabetical order)

### A

**ASD**

Agile Software Development. An approach to software development under which requirements and solutions evolve through the collaborative effort of self-organizing and small highly motivated team. It advocates continual improvement and encourages rapid and flexible response to change.

### F

**FR**

Functional Requirements. FR are the working characteristics of a product. These are based on how end users will use the product.

### N

**NFR**

Non-Functional Requirements. NFRs define system attributes such as security, reliability, performance, maintainability, scalability etc.

**NLTK**

The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs for symbolic and statistical natural language processing for English written in the Python programming language.

### O

**OOP**

Object Oriented Programming. OOP is a computer programming model that organizes software design around data, or objects, rather than functions and logic.