“IoT Based Smart Garbage Monitoring System”

A project report submitted in partial fulfillment of
requirements for the award of degree of

BACHELOR OF COMPUTER APPLICATIONS

Submitted By:

Mr. Suraj Harogoppa **USN: M2011251**

Under the Guidance of:

Prof. Bharati Ambali

BHARATESH COLLEGE OF COMPUTER APPLICATIONS

Affiliated to Rani Channamma University, Belagavi

2022-23

Bharatesh Education Trust's

BHARATESH COLLEGE OF COMPUTER APPLICATIONS

B.C. 188, Dharwad Road, Belagavi – 590 016

Certificate

This is to certify that the project entitled **"IOT BASED SMART GARBAGE MONITORING SYSTEM"** was satisfactorily completed by **Mr. Suraj Harogoppa.** **(M2011251).**under the partial fulfillment of requirement of the degree of Bachelor of Computer Applications under Rani Channamma University, Belagavi from APRIL 2022 to AUGUST 2023.

Internal Guide	**Principal**
Prof. Bharati Ambali	**Dr. Mahesh Mardolkar**

Examiners:

1.____________________________

2.____________________________

ACKNOWLEDGEMENT

It is a great pleasure to acknowledge all those who have contributed towards the conception, origin and nurturing of this project "Emo Player". We would like to extend thanks to many people, who so generously contributed to the work presented in this report and like to place on record our heartfelt thanks to each one of them who helped us in this project whichever and whenever needed.

We are grateful our enthusiastic principal**, Dr. Mahesh Mardolkar,** for active interest he showed in our project and similar, profound gratitude goes to our internal guide **Prof. Bharati Ambali** for guiding us in the project with her suggestions, co-operation and encouragement throughout our project.

Special mention goes to all the faculties of BCCA, our friends for giving immense support and motivation. Finally, thanks go to our parents for their support and yes, the almighty God abetted us a lot to complete this project successfully.

With Heartfelt Thanks to One and All

Mr. Suraj Harogoppa.

SYNOPSIS

INTRODUCTION

One of the main concerns with our environment has been solid waste management which impacts the health and environment of our society. The detection, monitoring and management of wastes is one of the primary problems of the present era. The traditional way of manually monitoring the wastes in waste bins is a cumbersome process and utilizes more human effort, time and cost which can easily be avoided with our present technologies. This is our solution, a method in which waste management is automated. This is our IoT Garbage Monitoring system, an innovative way that will help to keep the cities clean and healthy.

ABSTRACT

The overflowing of garbage is a sanitary issue which might cause diseases like cholera and dengue. Moreover it is a waste of fuel to travel around a complex or an area to find that some of the garbage are filled and some are not. Also, on rare days, problems might arise that there is so much garbage that the truck doesn't have enough capacity. Considering the above issues, this project proposes a system, that will try to reduce these problems to a greater extent. The system gives a real time indicator of the garbage level in a trashcan at any given time. Using that data we can then optimize waste collection routes and ultimately reduce fuel consumption. It allows trash collectors to plan their daily/weekly pick up schedule.

EXISTING SYSTEM

The existing system collects garbage once a day. It can also happen that sometimes waste spills out of the bins. Its unhygienic for the people too and leads to bad odor around the surrounding and leads to spreading some deadly diseases. To tackle such situation, a system is proposed named as "IoT based Garbage Monitoring System".

PROPOSED SYSTEM

The system uses an Ultrasonic Sensor which is used for detecting whether the trashcan is filled with garbage or not. Here Ultrasonic Sensor is installed at the top of Trash Can and will measure the distance of garbage from the top of Trashcan and we can set a threshold value according to the size of trash can. If the distance will be less than this threshold value, means that the Trashcan is full of garbage and we will print the message "Basket is Full" on the message and if the distance will be more than this threshold value, then we will print the distance remaining for the garbage vat to be full.

HARDWARE SPECIFICATIONS

1. HC-SR04 ultrasonic sensor.

2. Arduino Uno.

3. GSM module

4. Connecting wires.

5. DHT 11 pin out.

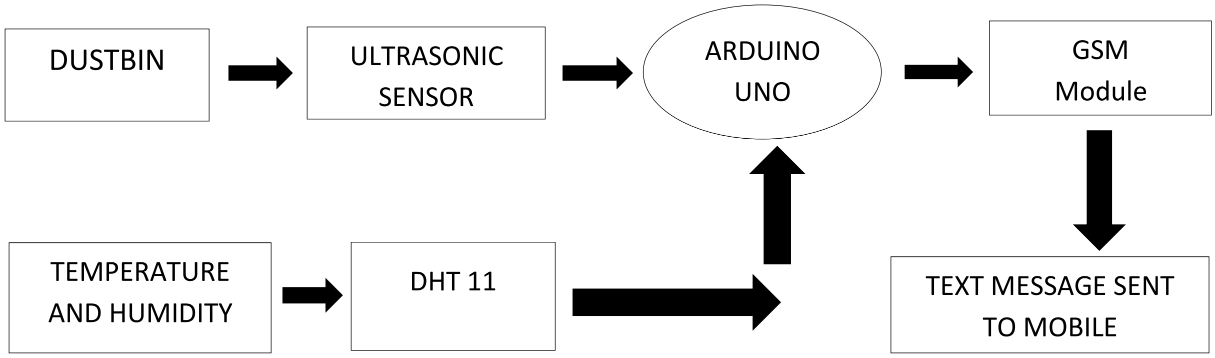
6. Node MCU.

SOFTWARE SPECIFICATIONS

1. Arduino IDE

2. Blynk app

DATAFLOW DIAGRAM



ADVANTAGES

1. Very simple circuit.

2. The HCSR04 sensor is very rugged.

3. Helps monitor garbage levels.

4. Uses very small amount of electricity

5. Ultimately helps in better planning of garbage pickups.

6. Can help in reducing overflowing bins.

7. Reduces trips to areas where the bins still have a lot of capacity.

DISADVANTAGES

1. Cannot detect liquid waste.
2. Only detects the top of the garbage level. It wouldn’t realize if there is space left.
3. GSM module needs a 12v source..

CONCLUSION

We built an efficient garbage monitoring system which can be used to monitor the level of garbage in the dump. This data can be further used to plan garbage collection trips more efficiently, ultimately reducing overflowing bins and helping have better public sanitation.

INDEX

INTRODUCTION

1. INTRODUCTION

We are living in an age where tasks and systems are fusing together with the power of IOT to have a more efficient system of working and to execute jobs quickly! With all the power at our finger tips this is what we have come up with. The Internet of Things (IoT) shall be able to incorporate transparently and seamlessly a large number of different systems, while providing data for millions of people to use and capitalize. Building a general architecture for the IoT is hence a very complex task, mainly because of the extremely large variety of devices, link layer technologies, and services that may be involved in such a system. One of the main concerns with our environment has been solid waste management which impacts the health and environment of our society. The detection, monitoring and management of wastes is one of the primary problems of the present era. The traditional way of manually monitoring the wastes in waste bins is a cumbersome process and utilizes more human effort, time and cost which can easily be avoided with our present technologies. This is our solution, a method in which waste management is automated. This is our IoT Garbage Monitoring system, an innovative way that will help to keep the cities clean and healthy.

OBJECTIVE AND SCOPE OF SYSTEM

2. OBJECTIVE AND SCOPE OF SYSTEM

The primary objective of the IoT-Based Garbage Monitoring System is to provide timely and accurate information about the fill-level of garbage bins throughout a city or urban area. It employs ultrasonic sensors installed within the bins to measure the fill-level and transmit the data wirelessly to a centralized platform.

This project provides real-time data on garbage levels to optimize waste collection routes and schedules which reduces unnecessary trips and fuel consumption by garbage collection trucks.

Scope for the project IoT based Smart Garbage Monitoring System:

- Waste management authorities can optimize their collection routes, ensuring that bins are emptied only when needed, thus saving time, fuel, and resources. By optimizing waste collection routes and schedules based on real-time data, the system helps reduce operational costs for waste management authorities. It eliminates the need for regular, fixed-time collection rounds and allows resources to be allocated more efficiently.

- Smart garbage monitoring systems help ensure that garbage bins are not overflowing, minimizing littering and maintaining cleanliness in public spaces. This leads to improved hygiene and reduces the risk of attracting pests and diseases.

- The system generates valuable data on waste generation patterns, bin usage, and fill levels. Waste management authorities can analyze this data to identify trends, make informed decisions about waste collection strategies, and plan future infrastructure requirements.

  Overall, the scope for an IoT-based smart garbage monitoring system is extensive, ranging from operational efficiency and cost reduction to environmental sustainability and citizen engagement. By leveraging real-time data and connectivity, such systems have the potential to revolutionize waste management practices and contribute to smarter, cleaner, and more sustainable cities.

THEORETICAL BACKGROUND
DEFINITION OF PROBLEM

3. THEORETICAL BACKGROUND DEFINITION OF PROBLEM

3.1 Theoretical Background:

Traditionally, waste management processes have relied on fixed schedules or manual inspections to determine when garbage bins need to be emptied. However, these methods are often inefficient and result in unnecessary costs and inconvenience. The smart garbage monitoring system addresses these challenges by providing real-time data on the fill levels of bins, enabling optimized waste collection and improved resource allocation.

The theoretical background of an IoT-based smart garbage monitoring system combines concepts from IoT, sensor technologies, and waste management principles to create a connected and intelligent waste management solution.

3.2 Drawback in Existing System:

- Traditional waste management systems typically follow fixed schedules for garbage collection, irrespective of the actual fill levels of the bins. This can result in unnecessary trips to empty bins that are not yet full, leading to wastage of time, fuel, and resources.
- Without real-time monitoring, garbage bins may overflow, leading to littering and unhygienic conditions. This can attract pests, cause foul odors, and create an unsightly environment.
- Inefficient waste collection routes and schedules contribute to increased carbon emissions and energy consumption. Traditional systems may not prioritize environmental sustainability and fail to optimize waste management operations from an ecological perspective.

3.3 Proposed System:

The system uses an Ultrasonic Sensor which is used for detecting whether the trashcan is filled with garbage or not. Here Ultrasonic Sensor is installed at the top of Trash Can and will measure the distance of garbage from the top of Trashcan and we can set a threshold value according to the size of trash can. If the distance will be less than this threshold value, means that the Trashcan is full of garbage and we will print the message “Basket is Full” on the message.

3.4 Advantages of Proposed Statement:

- Very simple circuit.
- The HCSR04 sensor is very rugged.
- Helps monitor garbage levels.
- Uses very small amount of electricity.
- Ultimately helps in better planning of garbage pickups.
- Can help in reducing overflowing bins.
- Reduces trips to areas where the bins still have a lot of capacity.

SYSTEM ANALYSIS AND DESIGN

4. SYSTEM ANALYSIS AND DESIGN

4.1 System Analysis:

System Analysis is the detailed study of the various operations performed by the system and their relationships within and outside the system. Analysis is the process of breaking something into its parts so that the whole may be understood. System analysis is concerned with becoming aware of the problem, identifying the relevant and most decisional variables, analyzing and synthesizing the various factors and determining an optimal or at least a satisfactory solution. During this a problem is identified, alternate system solutions are studied and recommendations are made about committing the resources used to design the system.

4.1.1 Analysis of IoT based Smart Garbage Monitoring System:

a) Functionality:

- The system continuously monitors the fill levels of garbage bins in real-time using sensors. This functionality ensures that up-to-date data is collected to facilitate efficient waste management.
- The system sets predefined fill level thresholds for garbage bins and generates alerts or notifications to waste management authorities when the fill levels exceed or reach these thresholds. This ensures timely waste collection and prevents overflow.

b) Efficiency:

The efficiency of an IoT-based smart garbage monitoring system lies in its ability to optimize resource allocation, reduce operational costs, improve waste collection processes, and foster citizen engagement. By leveraging real-time data, advanced analytics, and automation, the system streamlines waste management operations, leading to significant efficiency gains.

c) Scalability and Flexibility:

IoT-based systems can scale up to accommodate a large number of garbage bins and adapt to changing waste management requirements. As the system grows, it can efficiently handle increased data volumes and optimize operations across a wider area, ensuring scalability and flexibility.

4.2 Feasibility Study:

A feasibility analysis usually involves a through assessment of the operational (need), financial and technical aspects of a proposal. Feasibility study is the test of the system proposal made to identify whether the user needs may be satisfied using the current software and hardware technologies, whether the system will be cost effective from a business point of view and whether it can be developed with the given budgetary constraints. A feasibility study should be relatively cheap and done at the earliest possible time. When a new project is proposed, it normally goes through feasibility assessment. Feasibility study is carried out to determine whether the proposed system is possible to develop with available resources and what should be the cost consideration. Facts considered in the feasibility analysis were.

a) Economical Feasibility: An IoT-based smart garbage monitoring system can offer several economic benefits, making it economically feasible for waste management authorities, municipalities, and service providers. Here are some factors that contribute to its economic feasibility:

- **Efficient Resource Allocation:** Waste management authorities can plan collection routes based on actual fill levels, reducing unnecessary pickups and optimizing fuel consumption and labor costs.
- **Reduced Operational Costs:** By monitoring garbage levels, the system can schedule collections based on demand, resulting in reduced operational costs, such as fuel, labor, and vehicle maintenance expenses.
- **Preventing Overflow and Littering:** Overflowing bins can lead to littering and additional cleanup costs. The smart garbage monitoring system can send alerts when bins are nearing capacity, enabling timely collection and reducing the chances of overflow. This helps avoid the costs associated with cleaning up littered areas.

b) Operational Feasibility:

By offering real-time monitoring, automated alerts, centralized data management, scalability, and integration capabilities, Analytics and Reporting, Energy Efficiency an IoT-based smart garbage monitoring system ensures operational feasibility and enhances overall waste management efficiency.

c) Technical Feasibility: An IoT-based smart garbage monitoring system is technically feasible by leveraging various technologies and components such as Sensors like ultrasonic sensors, microcontrollers like Arduino, connectivity options like WiFi, cellular networks (2G,3G,4G), data transmission using MQTT or HTTP protocols, stored in cloud platform or server.

The smart garbage monitoring system can be integrated with existing backend systems used by waste management authorities or service providers. This integration enables seamless data flow and facilitates automated workflows, such as scheduling collection routes or generating work orders.

4.3 System Design

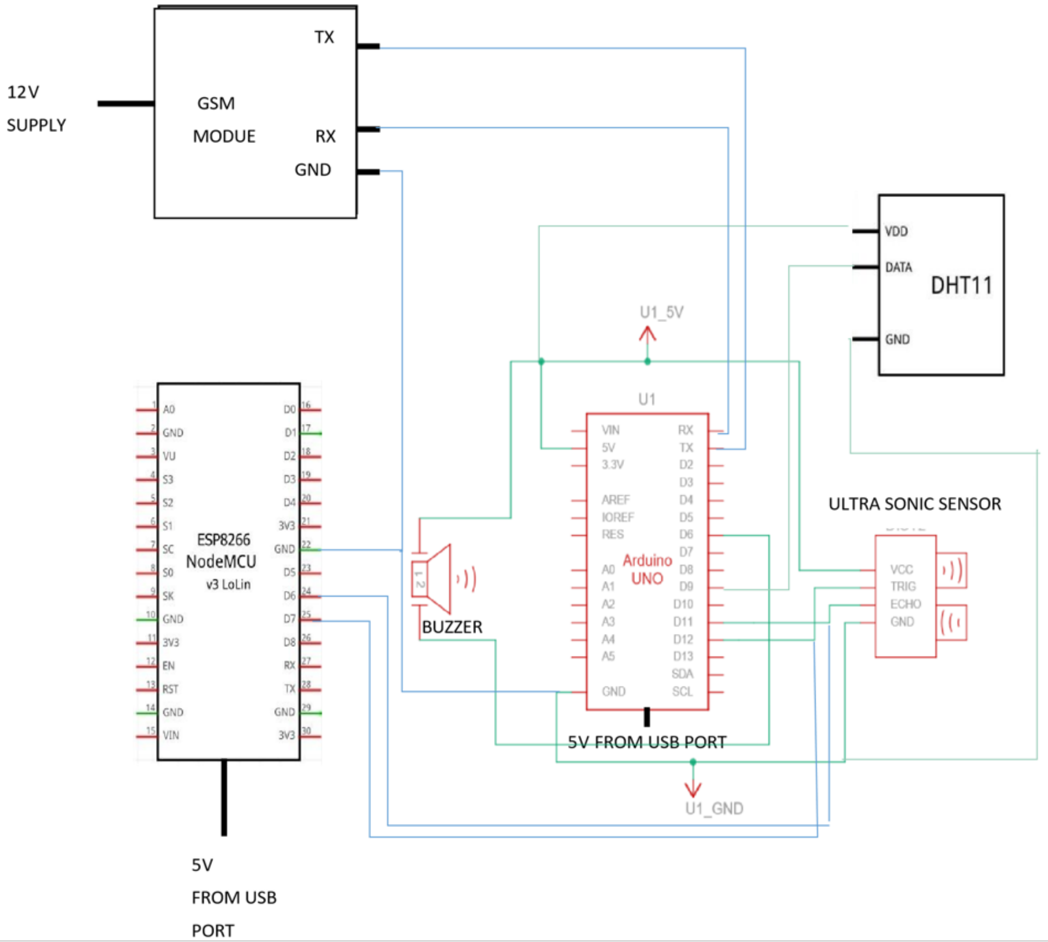


Fig:4.1 Circuit Diagram

USER REQUIREMENTS AND SYSTEM PLANNING

5.USER REQUIREMENTS AND SYSTEM PLANNING

5.1 User Requirements:

A good set of requirements are needed for any project, especially computer system projects, to be successful. This is where many projects fail, in that they do not specify correctly what the system should do.

5.1.1 Requirements Definition:

The truth is that you do not need a great deal of technical knowledge to specify requirements. In fact, it can be big disadvantages. A requirement for a computer system specifies what you want to desire from a system

- Functional Requirements
- Non-Functional requirements

a) Functional requirements:

- **Garbage Level Monitoring:** The system is able to monitor the fill level of garbage bins in real-time. This can be achieved through various sensors such as ultrasonic sensors. The system should accurately measure and report the fill level of each garbage bin.

- **Data Transmission:** The system has the capability to transmit the garbage level data from the sensors to a central server or cloud-based platform. The communication can be established using wireless protocols like Wi-Fi, Bluetooth, or Low-Power Wide-Area Network (LPWAN) technologies.

- **Alerts and Notifications**: The system provides alerts and notifications when the garbage level reaches a predefined threshold. This can help in timely waste collection and optimize the garbage collection routes. The alerts can be sent to waste management personnel, supervisors, or even automated systems responsible for waste collection.

- **Remote Monitoring and Control:** The system allows remote monitoring and control of the garbage bins. This includes functionalities like remotely checking the fill level, status monitoring, configuring thresholds, and adjusting settings of the sensors or communication modules.

b) Non-Functional Requirements:

- Performance
- Security
- Scalability
- Reliability
- User Experience

c) Gant Chart:

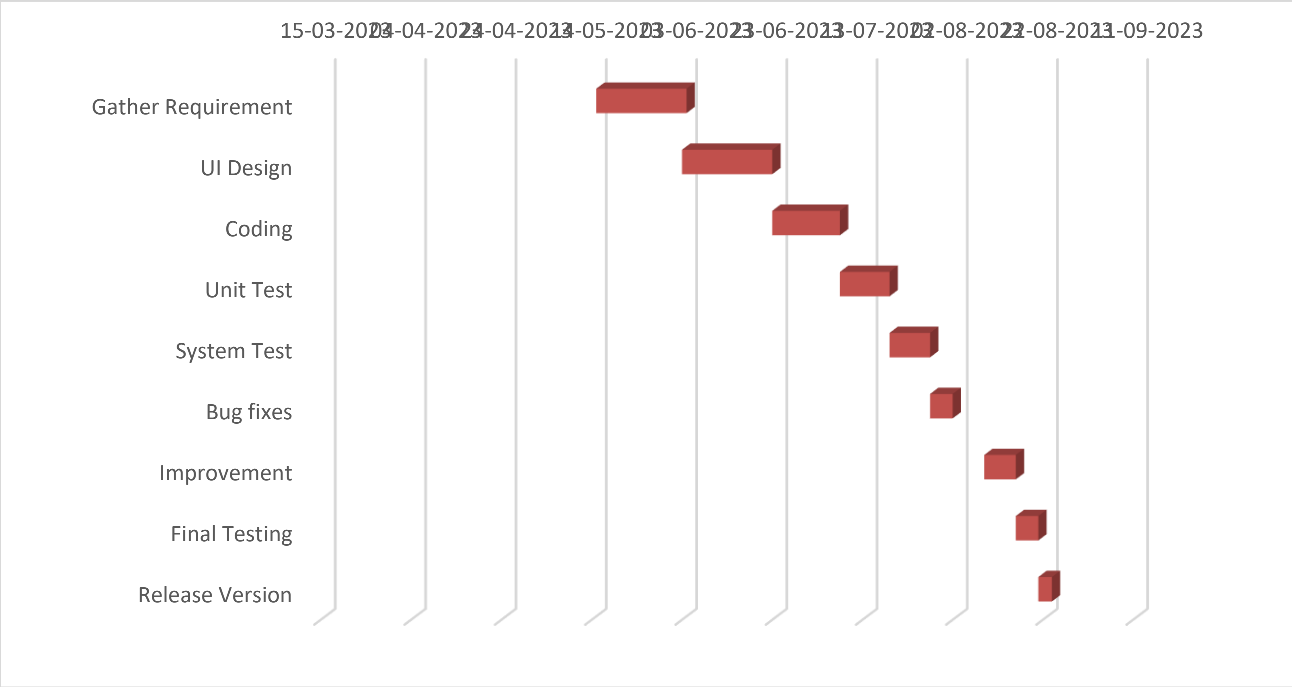


Fig:5.1 Gant Chart

5.2 System Planning:

System planning should be affective so that the project begins with well-defined task. Affective project planning helps to minimize the additional costs incurred while it is in progress. For the effective system planning, some principles are followed. These principles are listed below.

- **Planning is necessary:** Planning should be done before a project begins. For effective planning, objectives and schedules should be clear and understandable.

- **Risk Analysis:** Before starting the project, senior management and the project management team should consider the risk that may affect the system. For example: the user may desire the changes in requirement while the project is in progress in such a case the estimation of time should done according to those requirements.

- **Tracking of project plan:** Once the project plan is prepared, it should be tracked and modified accordingly.

- **Meet quality standards and produce quality deliverables:** The project plan should identify processes by which the project management team can ensure the quality in software, based on the process selected for ensuring quality, time for the project is estimated.

- **Description of flexibility to accommodate a change:** The result of project planning is recorded the form of a project plan, which should allow new changes to be accommodate when the project is in progress.

METHODOLOGY ADOPTED

6. METHODOLOGY ADOPTED

A software development methodology is a framework that is used to structure, plan, and control the process of developing an information system, this includes the pre-definition of specific deliverables and artefacts that are created and completed by a project team to develop or maintain an application. A wide variety of such frameworks have evolved over the years, each with its own recognized strengths and weakness. Each of the available methodology frameworks are best suited to specific kinds of projects, based on various technical, organizational, project and team considerations. The methodology framework is often defined in some kind of formal documentation. To implement the project goals, the following

Methodologies need to be followed:

- Specifying the Application and various components of the Architecture.
- Specifying the bindings between the tasks and the resources either manually or by the design tools.
- Specifying the port interconnections between the resources.
- Analysis, extracting the data required for analysis and the doing the analysis.

6.1 Software process Used:

6.1.1 Spiral model:

The spiral model combines the idea of iterative development with the systematic, controlled aspects of the waterfall model. This Spiral model is a combination of iterative development process model and sequential linear development model i.e. the waterfall model with a very high emphasis on risk analysis. It allows incremental releases of the product or incremental refinement through each iteration around the spiral.

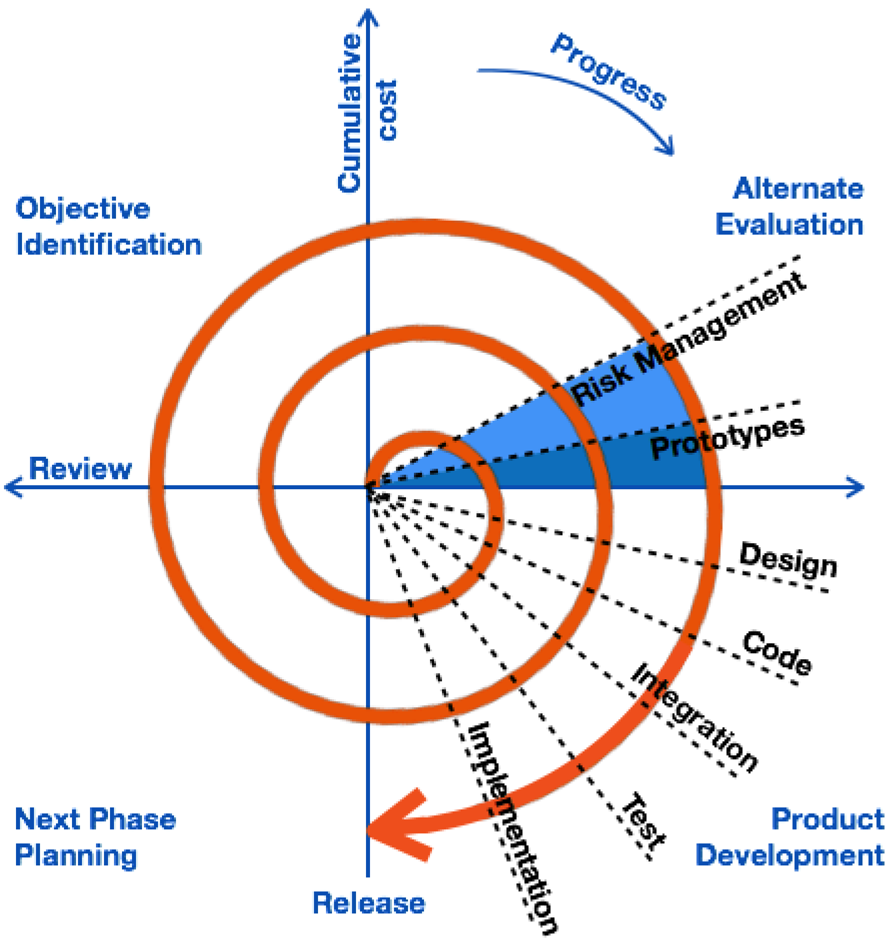


Fig:6.1 Spiral Model

- **Identification**

This phase starts with gathering the business requirements in the baseline spiral. In the subsequent spirals as the product matures, identification of system requirements, subsystem requirements and unit requirements are all done in this phase.

This phase also includes understanding the system requirements by continuous communication between the customer and the system analyst. At the end of the spiral, the product is deployed in the identified market.

- **Design**

The Design phase starts with the conceptual design in the baseline spiral and involves architectural design, logical design of modules, physical product design and the final design in the subsequent spirals.

- **Construct or Build**

The Construct phase refers to production of the actual software product at every spiral. In the baseline spiral, when the product is just thought of and the design is being developed a POC (Proof of Concept) is developed in this phase to get customer feedback.

Then in the subsequent spirals with higher clarity on requirements and design details a working model of the software called build is produced with a version number. These builds are sent to the customer for feedback.

- **Evaluation and Risk Analysis**

Risk Analysis includes identifying, estimating and monitoring the technical feasibility and management risks, such as schedule slippage and cost overrun. After testing the build, at the end of first iteration, the customer evaluates the software and provides feedback.

6.2 Software Description:

6.2.1 Arduino IDE

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software. This software can be used with any Arduino board. The Arduino development environment contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions, and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them. Software written using Arduino are called sketches. These sketches are written in the text editor. Sketches are saved with the file extension .ino. It has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino environment including complete error messages and other information. The bottom right-hand corner of the window displays the current board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

6.2.1.1 Blynk App

Blynk is a low-code IoT software platform for connecting devices to the cloud, building mobile apps to remotely control and monitor them, and managing thousands of users and deployed products. It’s a PaaS (Platform-as-a-Service) that helps businesses and individuals seamlessly progress from a prototype of a connected product to its commercial launch and further growth. All Blynk plans include native mobile apps, in addition to all of the other typical IoT infrastructure. With over 400 hardware models support, customers can connect any device to the Internet and use a suite of software products to run commercial projects.

Blynk Library is an extension that runs on your hardware. It handles connectivity, device authentication in the cloud, and commands processing between Blynk app, Cloud, and hardware. It's highly flexible whether you are starting from scratch, or integrating Blynk into existing project.

Blynk.Console is a feature-rich web application catering to different types of users. Its key functionalities include:

1. Configuration of connected devices on the platform, including application settings.
2. Device, data, user, organization, and location management.
3. Remote monitoring and control of devices.

Blynk.Cloud is a server infrastructure acting as the heart of Blynk IoT platform binding all the components together. Blynk also offers private servers. Blynk is a multi-tenant solution that allows you to configure user access to devices and data by defining roles and permissions.

SYSTEM IMPLEMENTATION & DETAILS OF HARDWARE & SOFTWARE USED

7. SYSTEM IMPLEMENTATION & DETAILS OF HARDWARE & SOFTWARE USED.

Implementation is the stage in the project where the theoretical design is turned into a working system and is giving confidence on the new system for the users, which it will work efficiently and effectively. It involves careful planning, investigation of the current system and its constraints on implementation, design of methods to achieve the change over an evolution, of change over methods. Apart of planning major task of preparing the implementation are education and training of users. The more complex system being implemented, the more involved will be the system analysis and the design effort required just for implementation.

7.1 HARDWARE & SOFTWARE REQUIREMENT

Hardware specifications:

- HC-SR04 ultrasonic sensor.
- Arduino Uno.
- GSM module
- Connecting wires.
- DHT 11 pin out.
- Node MCU.

Software Specifications:

- Blynk app
- Arduino IDE

SYSTEM MAINTENANCE & EVALUATION

8. SYSTEM MAINTENANCE & EVALUATION

8.1 System Maintenance:

The results obtained from the evaluation process help the organization to determine whether its information systems are effective and efficient or otherwise. The process of monitoring evaluating and modifying of existing information systems to make required or desirable improvements may be termed as System Maintenance.

System Maintenance is an ongoing activity, which covers a wide variety of activities, including removing program and design errors, updating documentation and test data and updating user support. For the purpose of convenience, maintenance may be categorized into three classes, namely: Corrective Maintenance, Adaptive Maintenance, and Perfective Maintenance.

a) **Corrective Maintenance**: This type of maintenance implies removing errors in a program, which might have crept in the system due to faulty design or wrong assumptions. Thus, in corrective maintenance, processing or performance failures are repaired.

b**) Adaptive Maintenance**: In adaptive maintenance program functions are changed to enable the information system to satisfy the information needs of the user. This type of maintenance may become necessary because of organizational changes which may include:

 Change in information needs of managers.

 Change in system controls and security needs etc.

c) **Perfective Maintenance**: Perfective Maintenance means adding new programs or modifying the existing programs to enhance the performance of the information system. This type of Maintenance undertaken to respond to user’s additional needs which may be due to the changes within or outside of the organization. Outside changes are primarily environmental changes, which may in the absence of system maintenance; render the information system in effective and inefficient. These environmental changes include:

Changes in governmental policies, laws, etc.

Economic and competitive conditions.

New technology.

8.2 Evaluation:

8.2.1 Hardware Evaluation Factors:

When we evaluate computer hardware, we should first investigate specific physical and performance characters for each hardware component to be acquired. These specific questions must be answered concerning many important factors. These hardware evaluation factors are: Performance, Cost, Reliability, Availability, Compatibility, Modularity, Technology, Connectivity, Environmental requirements, Software. There is much more to evaluating hardware then determining the fastest and cheapest computing device. For example, the question of possible obsolescence much be addressed by making a technology evaluation. The factor of ergonomics is also very important.

8.3 Cost and Benefit analysis:

Cost Analysis:

Cost of each Component:

- Arduino UNO : Rs.1,160
- DHT11 Sensor : Rs.120
- Ultrasonic Sensor : Rs.350
- GSM Module : Rs.900
- Battery : Rs.100
- Node MCU : Rs 250
- Connecting Wire : Rs 50
- Development cost : 10450
- Total : Rs.13380

The Constructive Cost Model (COCOMO) is an algorithmic software cost estimation model developed by Barry Boehm. The model uses a basic regression formula with parameters that are derived from historical project data and current as well as future project characteristics.

In detailed COCOMO, the effort is calculated as function of program size and a set of cost drivers given according to each phase of software life cycle. A Detailed project schedule is never static. The five phases of detailed COCOMO are: -

➢ Plan and requirement.

➢ System Design and Analysis.

➢ Implement code and test.

➢ Integration and test.

Requirement Gathering and Analysis = 20 days

System Design and Analysis = 20 days

Implement Code and Test = 35 days

Integration and Test = 5 days

Improvement and Final testing = 15

Effort cost Estimation = 110 Rs/day

Total Cost of the Project = 95 * 110 = 10,450 rupees

Data Flow Diagram

9. Data Flow Diagram

A data flow diagram is graphical tool used to describe and analyse movement of data through a system. These are the central tool and the basis from which the other components are Developed. The transformation of data from input to output, through processed. May be described logically and independently of physical components associated with the system. These are known as the logical data flow diagram in the DFD. There are four Symbols,

- A square defines a source (originator) or designation of system data.
- An arrow identifies data flow. It is the pipeline through which the information flows.
- A circle or a bubble represents a process that transforms incoming data flow into outgoing data flows.

9.1 Level 0 DFD:

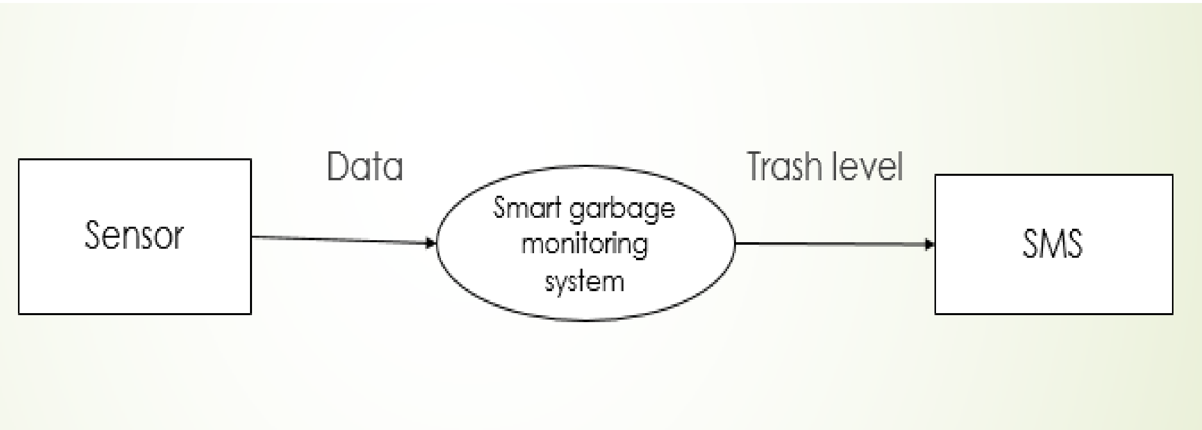


Fig: 9.1 Level 0 DFD

SOURCE-CODE

10. SOURCE-CODE

```
#include <SoftwareSerial.h>
#include <DHT.h>
#define DHTPIN 2
#define DHTTYPE DHT11
int buzzer = 6; // Buzzer connected to Arduino pin 6
DHT dht(DHTPIN, DHTTYPE);
SoftwareSerial mySerial(9, 10); // RX, TX for GSM module
int trigPin = 12;
int echoPin = 11;
int distance;
float humidity;
float temperature;

void setup() {
  pinMode(buzzer, OUTPUT); // Set buzzer pin as output
  dht.begin(); // Start DHT11 sensor
  mySerial.begin(9600); // Start GSM module serial communication
  pinMode(trigPin, OUTPUT); // Set trigPin as output
  pinMode(echoPin, INPUT); // Set echoPin as input
  Serial.begin(9600); // Start Arduino serial communication
}


void loop() {
  // Clears the trigPin
```

```
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
// Sets the trigPin on HIGH state for 10 micro seconds
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds
long duration = pulseIn(echoPin, HIGH);

// Calculating the distance
distance = duration * 0.034 / 2;

// Read DHT11 sensor values
humidity = dht.readHumidity();
temperature = dht.readTemperature();

// Print distance, temperature, and humidity values to serial monitor
Serial.print("Distance (cm): ");
Serial.println(distance);
Serial.print("Humidity: ");
Serial.print(humidity);
Serial.print(" %, Temperature: ");
Serial.print(temperature);
Serial.println(" °C");

// Check if the distance is less than 5cm
```

```
  if (distance <= 5) {
    digitalWrite(buzzer, HIGH); // Activate the buzzer for 1 second
    delay(1000);
    digitalWrite(buzzer, LOW);
    Serial.print("Dustbin is full! Please Empty it . The dustbin is located at Bharatesh Education Trust,Fort Road,Belagavi ");

    sendSMS("Dustbin is full! Please Empty it . The dustbin is located at Bharatesh Education Trust,Fort Road,Belagavi .Follow the link to reach the dustbin:https://maps.app.goo.gl/2zLmounckfNAZUy39"); // Send SMS notification
  }

  if (distance ==10) {
    digitalWrite(buzzer, HIGH); // Activate the buzzer for 1 second
    delay(100);
    digitalWrite(buzzer, LOW);
    Serial.print("Dustbin is about to full!  . The dustbin is located at Bharatesh Education Trust,Fort Road,Belagavi. ");

    sendSMS("Dustbin is about to full!  . The dustbin is located at Bharatesh Education Trust,Fort Road,Belagavi ."); // Send SMS notification
  }
  delay(5000);
}

void sendSMS(String message) {
  mySerial.println("AT+CMGF=1"); // Set GSM module in text mode
  delay(1000);
```

```
  mySerial.println("AT+CMGS=\"+918548062933\""); // Replace with recipient's phone number
  delay(1000);
  mySerial.println(message); // Send SMS message
  delay(100);
  mySerial.println((char)26); // End SMS transmission
  delay(1000);
}
```

Node MCU.ino

```
#include <dummy.h>
// Comment out these lines if you don't intend to use Blynk templates
 #define BLYNK_TEMPLATE_ID "TMPL3AEf1BEba"
 #define BLYNK_TEMPLATE_NAME "Suraj"

#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <Ultrasonic.h> // Library for HC-SR04 Ultrasonic Sensor
#include <DHT.h>        // Library for DHT11 Sensor

char auth[] = "h7eFGwRjBNaxuY9Z-Km0gJM8SblfvL3t";  // Replace with your actual Blynk authorization token
char ssid[] = "suraj";
char pass[] = "88888888";

#define TRIGGER_PIN D5  // Pin connected to HC-SR04 trigger
#define ECHO_PIN D6     // Pin connected to HC-SR04 echo
#define DHT_PIN D7      // Pin connected to DHT11 data pin
Ultrasonic ultrasonic(TRIGGER_PIN, ECHO_PIN); // Create an Ultrasonic object
```

```
DHT dht(DHT_PIN, DHT11); // Create a DHT object


void setup()
{
  Serial.begin(9600);
  Blynk.begin(auth, ssid, pass);
  dht.begin(); // Initialize DHT sensor
}


void loop()
{
   Blynk.run();
// Read distance from ultrasonic sensor
  long distance = ultrasonic.read();
 Serial.print("Distance: ");
 Serial.println(distance);
  // Read temperature and humidity from DHT11 sensor
  float temperature = dht.readTemperature();
  Serial.print("Tempreture: ");
  Serial.println(temperature);
  float humidity = dht.readHumidity();


 Serial.print("Humidity: ");
Serial.println(humidity);
  // Check if any readouts are valid
  if (!isnan(temperature) && !isnan(humidity))
  {
    // Send data to Blynk app
```

```
    Blynk.virtualWrite(V1, temperature); // Virtual Pin V1 for temperature
    Blynk.virtualWrite(V2, humidity);    // Virtual Pin V2 for humidity
    Blynk.virtualWrite(V3, distance);    // Virtual Pin V3 for distance
  }
  delay(1000); // Delay to avoid flooding Blynk with data
}
```

SNAPSHOTS

11. SNAPSHOTS

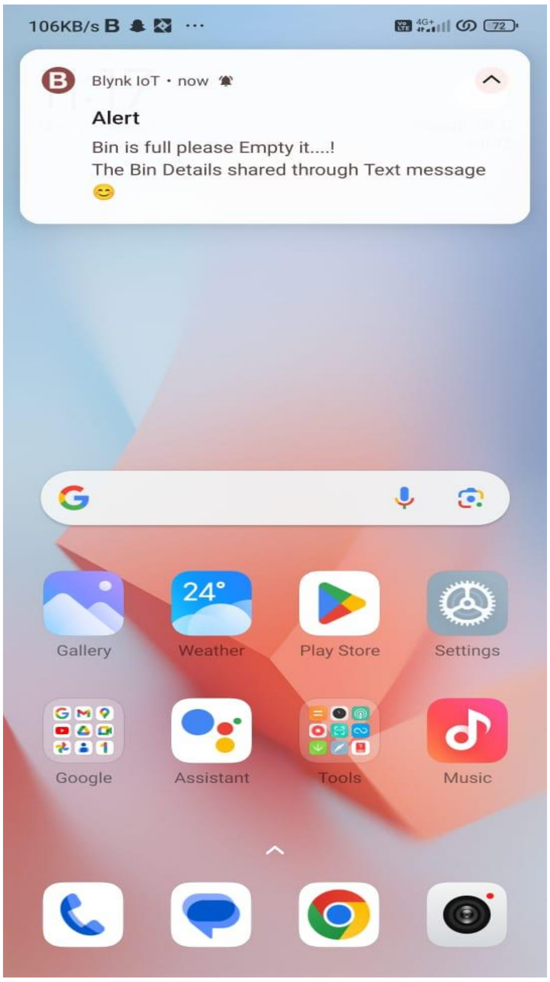
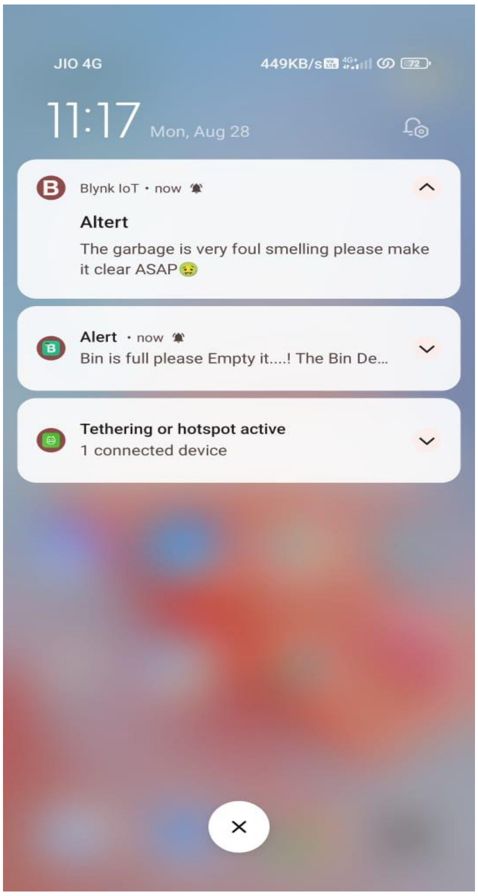


Fig:11.1 Mobile SMS



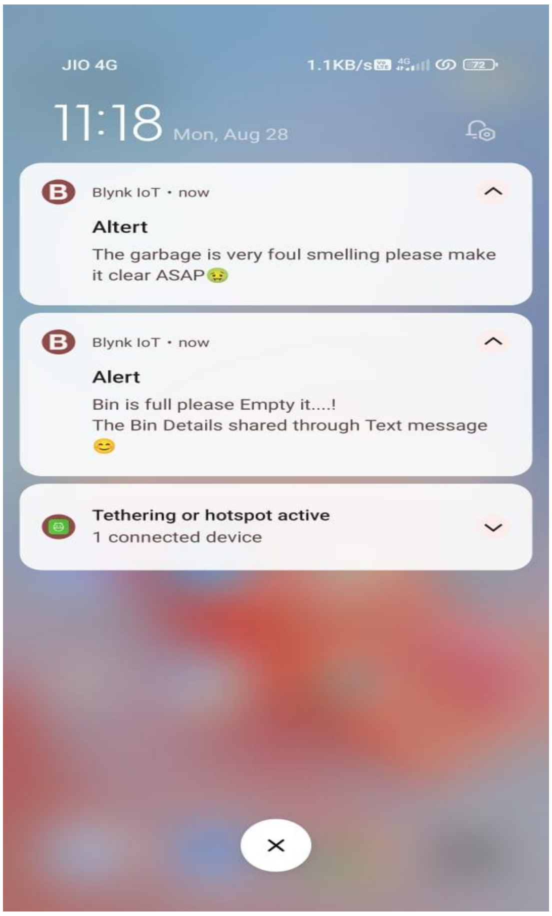


Fig:11.2 Blynk Notification

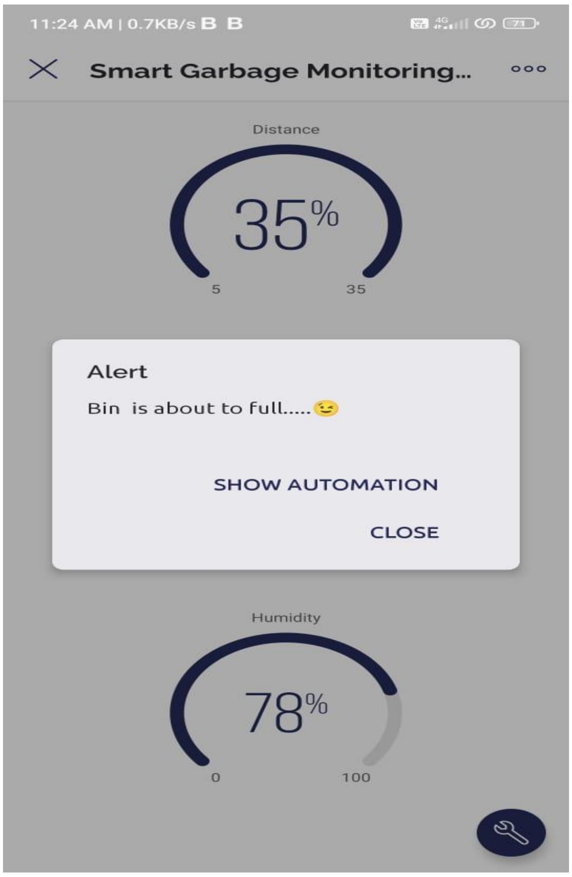


Fig:11.3 Blynk User Interface

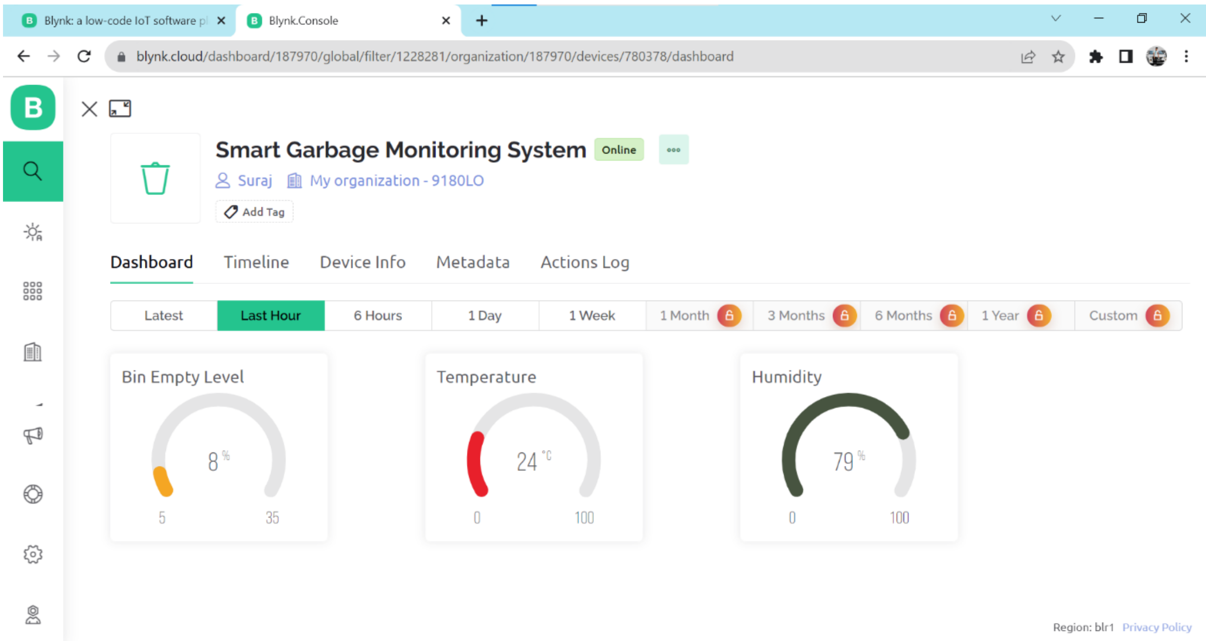


Fig:11.4 UI in Desktop

DEVICES USED

12 . DEVICES USED

12.1 Arduino UNO:

Fig: 12.1 Arduino Uno

Arduino is a microcontroller to control the working of the sensors and manage the working of the device. The Uno version of Arduino is implemented in this project. It was developed by Arduino CC. The Arduino board comes with various number of pins. The pins are categorized as output and input pins. The input pins accept digital as well as analog pins. It has 14 digital pins and 6 analog pins. It accepts 7 to 20 volts of power for working. It also has an USB port. The Uno was the first version of Arduino to be introduced in the Arduino family.

- **Function**: The Arduino Uno is a versatile development board designed for creating interactive and electronic projects. It's based on the ATmega328P microcontroller.
- **Features**: It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection for programming and power, and a power jack.
- **Programming**: Arduino Uno can be programmed using the Arduino IDE (Integrated Development Environment), a user-friendly platform that uses a simplified version of C/C++.
- **Usage**: It's widely used by hobbyists, students, and professionals to prototype and build various electronic projects, from simple LED blinking to complex robotic systems.
- **Interfacing**: You can connect various sensors, actuators, displays, and other components to the pins of the Arduino Uno, allowing it to interact with the physical world.
- **Ecosystem**: The Arduino platform has a vast community and large number of libraries and examples available online, making it easy to find support and resources for your projects.
- **Power**: The board can be powered via USB or an external power source. It can also provide power to external components.
- **Open Source**: Arduino Uno is an open-source platform, meaning its design and specifications are freely available for anyone to use and modify.
- **Compact:** Its compact size and ease of use make it suitable for beginners and experienced electronics enthusiasts alike.

12.2 HC-SR04 Ultrasonic Sensor

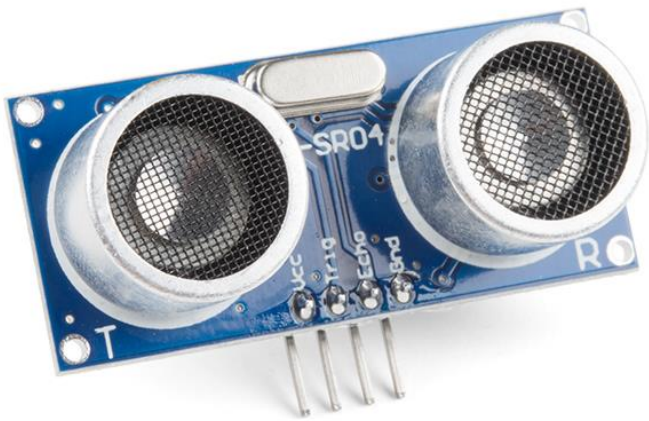


Fig: 12.2 HC-SR04 Ultrasonic Sensor

The HC-SR04 is an ultrasonic sensor module designed to measure distance by using ultrasonic sound waves. The acronym "HC-SR04" can be broken down as follows:

- HC: This doesn't have a specific meaning related to the sensor itself. It's likely a manufacturer code or identifier.
- SR04: This is the model or product code for the sensor module.

So, "HC-SR04" doesn't have a direct meaning beyond being a label or identifier for this particular ultrasonic distance sensor module. It's widely used and recognized in the maker and electronics communities as a specific type of ultrasonic sensor module with the characteristics and functionality described earlier.

12.3 GSM Module

Fig: 12.3 GSM Module

A GSM module, also known as a GSM modem or cellular module, is a hardware device that allows electronic devices to communicate over the cellular network. "GSM" stands for Global System for Mobile Communications, which is the standard used for mobile communication in most parts of the world. GSM modules are commonly used for sending and receiving SMS messages, making voice calls, and connecting to the internet via the cellular network. They are often integrated into various devices and systems to enable remote communication and control. These modules are particularly useful when Wi-Fi or wired internet connections are not available or practical. They are used in applications such as remote monitoring, security systems, tracking devices, and more.

12.4 DHT11

12.4 DHT11

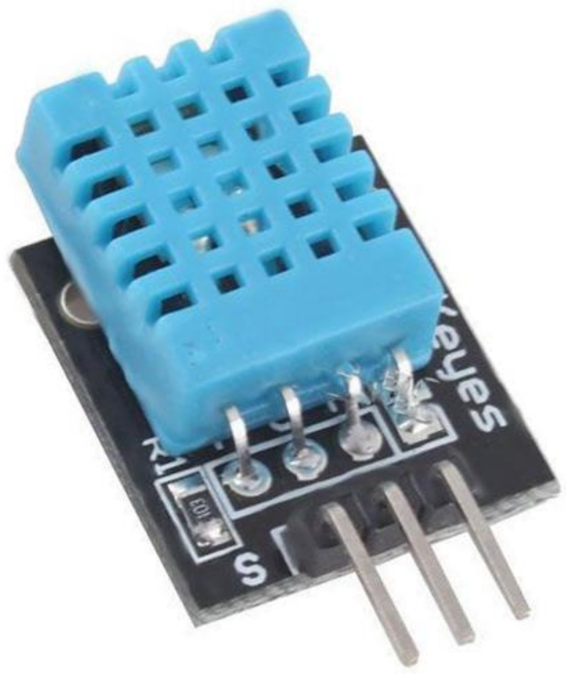


Fig:12.4 DHT11

The DHT11 is a popular digital temperature and humidity sensor module commonly used in electronics projects. Its pinout refers to the arrangement and function of its pins. Here's the pinout of the DHT11 sensor:

1. **VCC:** This pin is used to supply power to the sensor. It typically operates at 3.3V or 5V, depending on the sensor variant and the system's voltage levels.
2. **Data Out:** Also referred to as the "OUT" or "Signal" pin, this pin is used to transmit data from the sensor to a microcontroller. It carries both temperature and humidity data in a digital format.
3. **Not Connected:** Some DHT11 modules might have an extra pin that's not connected or used. This pin is often labeled as "NC" (No Connection) or left unmarked.
4. **Ground (GND)**: This pin is used to connect the sensor to the ground or 0V reference of the power supply.

12.5 Node MCU

Fig: 12.5 Node MCU

The NodeMCU is an open-source electronics platform based on the ESP8266 Wi-Fi module. It's designed to provide an easy way to prototype Internet of Things (IoT) projects, particularly those involving Wi-Fi connectivity.

Key Features:

- ESP8266 Chip: The NodeMCU board is built around the ESP8266 Wi-Fi module, which integrates a microcontroller with Wi-Fi capabilities.
- Wi-Fi Connectivity: NodeMCU enables devices to connect to Wi-Fi networks, making it suitable for IoT applications that require internet connectivity.
- Lua Scripting: The NodeMCU firmware initially used the Lua programming language for development, making it accessible for those familiar with scripting languages.
- Arduino Compatibility: Over time, the NodeMCU platform gained Arduino IDE compatibility, allowing users to program it using Arduino's familiar environment.
- GPIO Pins: NodeMCU features multiple General Purpose Input/Output (GPIO) pins that can be used to interface with sensors, actuators, and other components.
- USB Interface: The board often includes a USB-to-Serial converter, which simplifies programming and communication with computers.

- Expansion: There are various NodeMCU board versions and variants available, offering different features and configurations

PROCESS INVOLVED

13. PROCESS INVOLVED

Project process is also referred to as the development and normal life period of the project. It is the actual management of a project in practical terms. This is the level when the projects inputs are transformed into outputs via the project activities following the laid down work plan. This leads to the attainment of immediate objectives. It involves the coordinating, monitoring and control of the performance of the various project groups and the use of project resources in such a way that the project activities are completed in an orderly and optimal fashion within the constraint of time and resources available.

13.1 Approaches to Project Implementation:

13.1.1 Top-Down Approach:

The top-down approach for project implementation involves starting with a high-level perspective and gradually breaking down the project into smaller components and details. By following the top-down approach, you ensure that the project implementation process remains aligned with the overall goals and objectives of the IoT-based smart garbage monitoring system. It helps manage complexity by breaking down the project into manageable steps while maintaining a focus on the big picture.

13.1.2 Bottom-Up Approach:

The bottom-up approach for project implementation involves starting with individual components and gradually building up the system by integrating these components. By following the bottom-up approach, we build the IoT-based smart garbage monitoring system by gradually assembling and integrating individual components. This approach allows to focus on the technical aspects and interactions of the system while ensuring that each component functions correctly before integration.

METHODOLOGY USED FOR TESTING

14. METHODOLOGY USED FOR TESTING.

14.1 Type of Testing:

System Testing:

Testing phase is very important phase in system developing. No project is assumed complete until it is tested and elaborated test data is prepared and the system is tested using that test data.

- Testing is the process of executing a program with the intent of finding error.
- A good test case is the one high probability of finding as undiscovered error.

Unit Testing:

Unit testing is a crucial aspect of software development for an IoT-based smart garbage monitoring system. It involves testing individual components or units of code in isolation to ensure they work as intended. It involves breaking down the software codebase into smaller units, such as functions, classes, methods, or modules, that can be tested independently. In the smart garbage monitoring system, units could include data processing functions, communication modules, and sensor control logic.

For each unit, we create test cases that cover different scenarios and edge cases. These scenarios include valid inputs, invalid inputs, boundary cases, and error conditions. By performing thorough unit testing, we can identify and address issues early in the development process, leading to a more reliable and maintainable IoT-based smart garbage monitoring system.

Integration Testing:

Integration testing for an IoT-based smart garbage monitoring system involves testing the interactions and communication between various components, devices, and subsystems to ensure that they work together as a cohesive whole. In smart garbage monitoring system, integration points could include communication between sensors, microcontrollers, communication modules, central servers/cloud, and user interfaces.

We test the flow of data between different components and verify that data collected by sensors is accurately reflected in user interfaces. Ensuring that messages are correctly formatted, transmitted, and received, and that error handling mechanisms are effective. The system sends alerts or notifications to administrators, we test their delivery and accuracy and also verify that alerts are triggered correctly based on garbage fill levels.

Integration testing helps ensure that the different components of IoT-based smart garbage monitoring system function harmoniously together. By thoroughly testing the interactions and communication between components, we can identify and address integration issues early, leading to a more reliable and robust system.

Validation Testing:

Validation testing for an IoT-based smart garbage monitoring system involves verifying that the system meets the intended requirements. It focuses on ensuring that the system meets the needs of stakeholders and functions effectively within its deployment environment. It identifies key performance indicators (KPIs) that reflect the system's success in achieving its goals. So we measure the accuracy of garbage fill level predictions, the efficiency of waste collection routes, and the responsiveness of alert notifications.

14.2 Methodology used for testing:

Types of testing:

14.2.1 Black Box Testing

The technique of testing without having any knowledge of the interior workings of the application is Black Box testing. The tester is oblivious to the system architecture and does not have access to the source code. Typically, when performing a black box test, a tester will interact with the system's user interface by providing inputs and examining outputs without knowing how and where the inputs are worked upon.

Advantages	**Disadvantages**
Well suited and efficient for large code segments.	• Limited Coverage since only a selected number of test scenarios are actually performed.
• Code Access not required.	• Inefficient testing, due to the fact that the tester only has limited knowledge about an application.
• Clearly separates user's perspective from the developer's perspective through visibly defined roles.	• Blind Coverage, since the specific code segments or error prone areas.
• Large numbers of moderately skilled testers can test the application with no knowledge of implementation, programming language or operating systems	• The test cases are difficult to design.

14.2.2 White Box Testing

White box testing is the detailed investigation of internal logic and structure of the code. Box testing is also called glass testing or open box testing. In order to perform white box testing on an application, the tester needs to possess knowledge of the internal working of the code. The tester needs to have a look inside the source code and find out which unit/chunk of the code is behaving inappropriately. White-box testing is a method of testing the application at the level of the source code. These test cases are derived through the use of the design techniques mentioned above: control flow testing, data flow testing, branch testing, path testing, statement coverage and decision coverage as well as modified condition/decision coverage. White-box testing is the use of these techniques as guidelines to create an error-free environment by examining all code. These white-box testing techniques are the building blocks of white-box testing, whose essence is the careful testing of the application at the source code level to reduce hidden errors later on. These different techniques exercise every visible path of the source code to minimize errors and create an error-free environment. The whole point of white-box testing is the ability to know which line of the code is being executed and being able to identify what the correct output should be

Advantages	**Disadvantages**
• As the tester has knowledge of the source code, it becomes very easy to find out which type of data can help in testing the application effectively.	• Due to the fact that a skilled tester is needed to perform white box testing, the costs are increased.
• It helps in optimizing the code.	• Sometimes it is impossible to look into every nook and corner to find out hidden errors that may create problems as many paths will go untested.
• Extra lines of code can be removed which can bring in hidden defects	• It is difficult to maintain white box testing as the use of specialized tools like code analysers and debugging tools are required.

ANNEXURE

15. ANNEXURE

15.1 List of Abbreviations, Figures and Tables:

15.1.1 Abbreviations:

- DFD – Data flow Diagram
- IOT – Internet of Things
- MQTT – Message Quering Telemetry Transport
- HTTP – Hyper Text Transfer Protocol
- DHT 11 – Digital Humidity and Temperature sensor.

15.1.2 Figures:

15.2 References

Text Book:

1. ArshdeepBahga, Vijay Madisetti, ―Internet of Things – A hands-on approach, Universities Press, 2015

2. Olivier Hersent, David Boswarthick, Omar Elloumi , ―The Internet of Things – Key applications and Protocols, Wiley, 2012 (for Unit2).

3. Jan Ho¨ ller, VlasiosTsiatsis , Catherine Mulligan, Stamatis , Karnouskos, Stefan Avesand. David Boyle, "From Machine-to-Machine to the Internet of Things – Introduction to a New Age of Intelligence", Elsevier, 2014.

4. Dieter Uckelmann, Mark Harrison, Michahelles, Florian (Eds), ―Architecting the Internet of Things, Springer,2011.

5. Michael Margolis, Arduino Cookbook, Recipes to Begin, Expand, and Enhance Your Projects, 2nd Edition, O‟Reilly Media,2011.