**IoT Course**

Capstone Project   
Final Report

For students (instructor review required)

ⓒ2023 SAMSUNG. All rights reserved.

Samsung Electronics Corporate Citizenship Office holds the copyright of this document.

This document is a literary property protected by copyright law so reprint and reproduction without permission are prohibited.

To use this document other than the curriculum of Samsung Innovation Campus, you must receive written consent from copyright holder.

|  |
| --- |
| **Smart waste management using fill-level detection sensors** |

(02/07/2025)

**Group 7**

Trần Khắc Nam

Lê Văn Chính

Dương Quang Vinh

Content

1. Introduction

1.1. Background Information

1.2. Motivation and Objective

1.3. Members and Role Assignments

1.4. Schedule and Milestones

2. Project Execution

2.1. IoT Service Model

2.2. Data Processing

2.3. Service Implementation

2.4. System Design

3. Results

3.1. Data Acquisition (Sensor, Actuator, Controller)

3.2. Network and Communication

3.3. Hardware Implementation

3.4. Data Visualization

3.5. Testing and Improvements

4. Projected Impact

4.1. Accomplishments and Benefits

4.2. Future Improvements

5. Team Member Review and Comment

6. Instructor Review and Commen

1. Introduction

1.1. Background Information

Waste is one of the most urgent environmental and urban management challenges globally today. Rapid urbanization and population growth lead to an ever-increasing amount of domestic waste, causing many serious consequences. Landfills are frequently overloaded, causing soil, water, and air pollution due to leachate leakage and toxic gas emissions. In addition, overflowing trash cans in residential areas not only degrade urban aesthetics and generate unpleasant odors but also serve as a source of disease transmission, directly affecting the health and quality of life of the community.

Besides, the problem of waste getting wet due to rain is also a major issue, causing pollution, unpleasant odors, and affecting waste treatment operations Traditional waste management and collection methods currently rely on fixed schedules. This leads to many limitations and inefficiencies:

* Waste of resources: Garbage collection vehicles still have to travel and operate on a fixed schedule, even when the bins are not full. This increases operating costs (fuel, labor), wastes resources, and increases emissions.
* Overflowing bins: Conversely, in areas with large amounts of waste or during peak times, bins can overflow before the scheduled collection, causing localized pollution and unsanitary conditions.
* Lack of data: The lack of real-time data on bin fill levels makes it difficult for management units to make optimal decisions regarding collection routes and frequency.
* In this context, Internet of Things (IoT) technology emerges as a promising solution to address these challenges. IoT enables physical objects to be equipped with sensors, software, and other technologies to connect and exchange data with other systems and devices over the internet. By integrating sensors into trash bins, IoT has the ability to provide real-time data on fill levels, thereby allowing for more automated and intelligent waste management, optimizing collection processes, reducing costs, and contributing to building a cleaner and more beautiful urban environment. 1.2. Motivation and Objective

**Motivation:**

* This project arises from a practical and urgent need for a smarter and more efficient waste management solution. The current state of traditional waste collection methods often leads to wasted resources when collection vehicles travel to half-empty bins, or conversely, results in overflowing bins, creating unsanitary conditions and degrading urban aesthetics. Furthermore, the problem of waste getting wet due to rain is also a major issue, causing pollution, unpleasant odors, and affecting waste treatment operations.
* These challenges not only impact the operating costs of management units but also negatively affect the environment and the quality of life of the community.
* With the strong development of sensor technology and the Internet of Things (IoT), applying these technologies to waste management is considered entirely feasible and can bring superior efficiency. IoT provides the ability to collect and transmit real-time data, while also allowing the system to automatically react to environmental conditions, forming the basis for optimizing collection processes and making smarter decisions.

**Objective:**

Based on the aforementioned motivations, the main objective of this project is to design and implement a smart and environmentally adaptive waste management system to achieve the following results:

* Monitor Fill Level: Design and implement a system to monitor the fill level of trash bins using ultrasonic sensors (or other suitable sensor types to be used during project implementation).
* Automatic Lid Control Based on Weather: Integrate a rain sensor to detect weather conditions and control a servo motor to automatically close the trash bin lid when it rains, aiming to protect waste from getting wet and safeguard internal components.
* Real-time Data Transmission: Ensure continuous and real-time transmission of trash bin fill level data to a Cloud Platform.
* Develop Intuitive Interface: Build an intuitive interface in the form of a web dashboard or mobile application to display the current status of all trash bins, fill levels, and lid status (open/closed), while also providing alerts and notifications when bins are nearly full or when the lid opens/closes automatically.
* Optimize Collection Routes: Analyze fill level data to optimize waste collection routes and schedules, thereby helping to save operating costs (fuel, labor) and resources.
* Enhance Urban Management Efficiency: Contribute to improving environmental hygiene, minimizing pollution, protecting waste from weather impacts, and enhancing the overall efficiency of urban waste management, moving towards a smarter and more sustainable city.

1.3. Members and Role Assignments

1. **Trần Khắc Nam:**

 Develop a web/mobile dashboard interface to display data and integrate maps.

 Manage databases and perform integration tasks with the cloud platform.

 Develop logic for the notification system when trash bins reach a full level.

 Support compiling and synthesizing report sections related to software and data.

1. **Lê Văn Chính:**

 Develop firmware for microcontrollers, including functions for reading data from sensors and transmitting data.

 Design and install system hardware, including sensors, microcontrollers, and power supplies.

 Conduct tests and calibrate sensors to ensure accuracy.

 Compile report sections related to hardware, firmware, and device test results.

1. Dương Quang Vinh

 Research and implement automatic lid control function

 Develop specialized firmware for rain sensors and servo motors

 Test and optimize the lid opening/closing system

 Assist in compiling and synthesizing report sections

1.4. Schedule and Milestones

**Schedule:**

* **Week 1:** Research and overall system design. This phase focuses on defining requirements, selecting hardware and software technologies (sensors, microcontrollers, InfluxDB, Grafana, Node-RED, Arduino IDE), and creating the system architecture diagram.
* **Week 2:** Component selection and procurement, and basic hardware assembly. Perform preliminary connections of sensors and microcontrollers.
* **Week 3:** Develop firmware for the microcontroller using Arduino IDE. Focus on reading data from sensors and transmitting data to Node-RED. Set up Node-RED to receive data and push it into InfluxDB.
* **Week 4:** Build the dashboard/user interface using Grafana. Create intuitive charts and widgets to display data from InfluxDB.
* **Week 5:** Integrate the entire system (hardware, Node-RED, InfluxDB, Grafana), conduct testing and debugging. Check end-to-end data flow.
* **Week 6:** Optimize system performance, finalize the report, and prepare the final presentation.

**Milestones:**

 **Completed system architecture design:** The overall system diagram is clearly defined, including hardware components (sensors, microcontrollers), embedded software (Arduino IDE), and data management/visualization tools (Node-RED, InfluxDB, Grafana).

 **Functional hardware prototype:** Sensors are capable of accurate data reading, and the microcontroller can send data via wireless communication.

 **Data successfully stored and displayed:** Data from the device is transmitted via Node-RED, stored in InfluxDB, and visually displayed on the Grafana Dashboard.

 **Complete and intuitive data display dashboard:** The Grafana Dashboard interface provides comprehensive and easy-to-understand information about waste fill levels and historical charts.

 **Stable system operation and successful testing:** The entire system operates continuously, reliably, and achieves the stated functional objectives after thorough testing.

2. Project Execution

2.1. IoT Service Model

The IoT service model of the smart waste management system is designed with a layered architecture, ensuring efficient data collection, transmission, processing, and visualization. The main layers include:

• Device Layer: This layer directly interacts with the physical environment.

* An ultrasonic sensor is used to measure the distance from the bin lid to the trash surface. This distance data is used to determine the bin's fill level.
* The ESP32-C6 microcontroller acts as the brain of the device, responsible for reading sensor data and performing basic calculations to determine the fill percentage.

• Connectivity Layer: This layer ensures reliable data transmission.

* The ESP32-C6 connects to the Internet via Wi-Fi.
* The MQTT (Message Queuing Telemetry Transport) protocol is used for data transmission. This lightweight protocol is well-suited for IoT devices with limited bandwidth, enabling efficient and resource-saving communication of fill level data.

• Cloud Layer: This layer handles receiving, storing, and processing data from the devices.

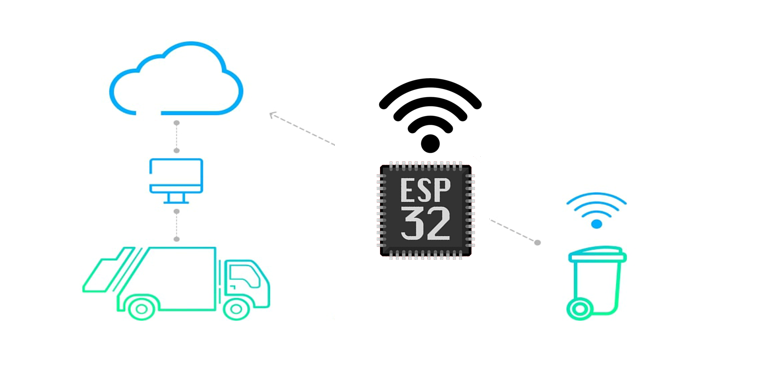
* The IoT cloud platform (including Node-RED connected to an MQTT Broker and InfluxDB) receives the fill level data from the devices.
* The data is stored in the real-time time-series database InfluxDB, and can be further processed (e.g., converting distance into accurate fill percentage, threshold detection).

• Application Layer: This layer provides an interactive interface and management functions for users.

* A web dashboard (developed using Grafana and Node-RED Dashboard) visualizes the fill level of each bin. The dashboard can also display historical fill level charts.
* The system can send automated alerts or notifications when a bin is nearing full capacity.

• Action Layer: Based on the data and analysis from the application layer, this layer triggers real-world actions.

* Notifications are sent to waste collection staff regarding full bin status.



2.2. Data Processing

The data processing in the system is carried out through multiple stages, from collection at the device to processing and storage on the cloud platform, ensuring that information about bin fill levels and environmental status is always updated and ready for analysis.

* **Data Collection at the Device** This layer focuses on collecting raw data from sensors and performing preliminary calculations as well as local control actions.
  + **Ultrasonic Sensor:** Used to measure the distance (in cm) from the top of the trash bin to the waste surface.
  + **Rain Sensor:** Reads signals to determine the presence of rain. This signal can be an analog value (humidity) or a digital value (raining/not raining).
  + **ESP32-C6 Microcontroller:** Plays a central role in:
  + Reading distance values from the ultrasonic sensor.
  + Reading values from the rain sensor.
  + **Controlling the bin lid:** Based on the signal from the rain sensor, the microcontroller will send control commands to the servo motor to close the bin lid when rain is detected and open it when the rain stops. This logic is implemented directly on the device to ensure a quick response.
  + **Data Conversion and Pre-processing:** The distance value measured from the ultrasonic sensor is converted into the percentage fill level of the trash bin using the formula:
  + Rain status is also recorded.
  + The lid status after the servo motor operates is also updated.
  + **Data Packaging:** The processed data (Fill Percentage, Rain Status, Lid Status) is then packaged into a JSON string and sent via MQTT under an appropriate topic
* **Sending Data to the Cloud Platform** The JSON data payload is transmitted to the cloud platform using the MQTT protocol, ensuring efficient and reliable data transfer, even in unstable network conditions or with limited bandwidth.
* **Data Processing and Storage on the Cloud** This layer is responsible for receiving, in-depth processing, storing, and triggering actions based on the received data.
* **Node-RED (Cloud Gateway):** Acts as a cloud gateway, receiving MQTT payloads from devices.
* **Data Parsing and Routing:** Node-RED performs intermediate tasks such as parsing JSON data, extracting values (Fill Percentage, Rain Status, Lid Status), and routing them to appropriate destinations.
* **Data Storage:** Data is then stored in InfluxDB, a time-series database chosen for its high efficiency in processing time-stamped sensor data.
* **Triggering Actions and Alerts:** Node-RED can be configured with logic flows (rule engines) to trigger automatic actions or alerts when:
  + - The trash bin's fill level exceeds a predefined threshold.
    - The rain status changes (e.g., from "not raining" to "raining" to log or send notifications).
    - The bin lid status changes (to monitor servo operation).
* **Building Visualization Data Flows:** Node-RED also builds data flows to feed Node-RED Dashboard (or other visualization tools like Grafana) for real-time data display.

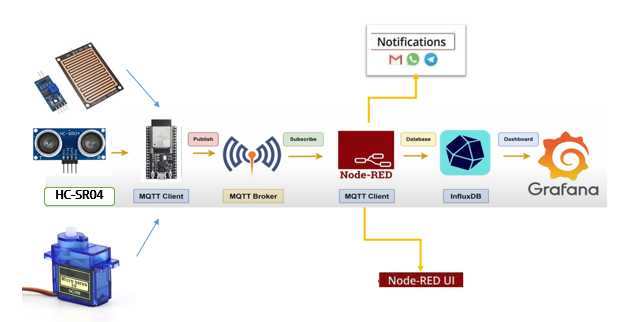
2.3. Service Implementation

The smart waste management system is built from a combination of hardware, firmware, a cloud platform, and a user interface, ensuring a comprehensive and effective solution.

* **Hardware** The main hardware components used to build the smart trash bin device include:
  + **Microcontroller:** ESP32-C6. This is the central control unit, responsible for processing data from sensors and controlling peripheral devices.
  + **Ultrasonic Sensor:** HC-SR04. Used to measure the distance from the top of the trash bin to the waste surface, thereby determining the fill level.
  + **Rain Sensor:** Detects the presence of rainwater to activate the lid protection mechanism.
  + **Servo Motor:** SG90. Responsible for automatically opening/closing the trash bin lid based on signals from the microcontroller, determined by the rain status.
  + **Power Supply:** 5V adapter (or a power supply suitable for the components; a voltage regulator may be needed if the servo requires higher current).
* **Firmware** Firmware is the program loaded into the microcontroller, managing all device operations.
  + **Programming Language:** C++.
  + **Integrated Development Environment (IDE):** Arduino IDE.
  + **Key Libraries:**
    - **WiFiManager:** Helps configure Wi-Fi connections easily and flexibly without hardcoding SSID and password.
    - **PubSubClient:** MQTT protocol library, allowing the device to send and receive data from the MQTT Broker on the cloud.
    - **Ultrasonic Sensor Library:** To easily read data from HC-SR04.
    - **Servo Library:** To control the servo motor accurately and efficiently.
  + **Operating Logic:**
    - Periodically read data from the ultrasonic sensor to calculate the bin's fill level.
    - Continuously read data from the rain sensor to monitor weather conditions.
    - **Control the servo motor:** Depending on the rain status, the microcontroller will command the servo motor to close or open the trash bin lid.
    - Manage Wi-Fi connection and connect to the MQTT Broker.
    - Package data (fill level, rain status, lid status) into JSON format and send it to the cloud via MQTT.
* **Cloud Platform** The cloud platform plays a central role in receiving, processing, and storing data from IoT devices.
  + **Node-RED:** Used to build data processing flows. Node-RED receives MQTT data from devices, parses JSON, and routes data to the database or alert systems. It also manages logic to trigger alerts based on fill level thresholds.
  + **InfluxDB:** A time-series database optimized for storing real-time sensor data. InfluxDB provides efficient data storage and retrieval for analysis and visualization.
* **User Interface** The user interface provides intuitive visualization and interaction for system managers.
  + **Dashboard Development Tool/Platform:** Node-RED Dashboard is the primary tool used to create the visual interface. Grafana can be used additionally for more in-depth and advanced analytical charts if needed.
  + **Data Display Description:** Node-RED Dashboard displays live data from Node-RED flows, which can be processed real-time data or data queried from InfluxDB. The main components of the dashboard include:
    - **Charts and Gauges:** Display the current fill level of each trash bin as a percentage.
    - **Lid Status Indicator:** Clearly displays the current status of the trash bin lid (open/closed).
    - **Rain Status Indicator:** Displays information about rain (present/absent) at the bin's location.
    - **Historical Charts:** Visualize fill level trends and lid open/close status over time.
    - **Alert System:** Displays notifications when a trash bin reaches a full threshold or when a significant event occurs.

2.4. System Design

* **Overall Architecture Diagram:**

****

**1. Smart Bin Unit**

* Ultrasonic Sensor (HC-SR04):  
  Measures the distance from the lid of the bin to the surface of the waste. This value is used to calculate the fill level as a percentage.
* ESP32-C6 Microcontroller:  
  Reads data from the ultrasonic sensor, calculates the fill level, and sends the data over Wi-Fi.
* Rain Sensor: YL-83 Rain Sensor Module. Detects the presence of rain, providing a signal to activate the mechanism for protecting waste from getting wet.
* Servo Motor**:** SG90 Micro Servo. Responsible for automatically closing and opening the trash bin lid based on signals from the rain sensor or other control commands.
* Built-in Wi-Fi Module:  
  Integrated in the ESP32-C6 to provide wireless Internet connectivity.
* Power Supply:  
  The system is powered by either batteries or an external adapter to ensure stable operation.

**2. Wireless Communication**

* Wi-Fi:  
  Connects the smart bin to the Internet.
* MQTT Protocol:  
  A lightweight publish/subscribe communication protocol used to transmit fill-level data to the cloud via an MQTT Broker.

**3. Cloud Platform**

* MQTT Broker:  
  Acts as the central hub for receiving data from multiple smart bins. Each bin publishes data to a specific MQTT topic.
* Node-RED:
  + Processes incoming data streams from MQTT topics.
  + Validates and transforms the sensor data as needed.
  + Forwards the processed data to InfluxDB for storage.
  + Triggers alerts (e.g., when a bin is full) using logic flows.
* InfluxDB:
  + A time-series database used to store fill-level data over **time.**
  + Enables historical analysis and data visualization.

**4. Application Layer & User Interface**

* Grafana**:**
  + Retrieves data from InfluxDB using custom queries.
  + Displays information using dashboards, including charts, gauges, and time-series graphs.
* Node-RED Dashboard:
  + Main web interface for real-time monitoring.
  + Displays key data using visual components like line charts, gauges, and status indicators.
  + Supports basic user interaction (e.g., buttons or toggle switches, if implemented).

**5. Action Layer (Optional but Supported)**

* + Automated Alerts:
  + Triggered when the fill level exceeds a predefined threshold.
  + Alerts may include:
    - Email notifications (e.g., via Gmail node in Node-RED)
    - LED flashing as a local visual indicator

3. Results

3.1. Data Acquisition (Sensor, Actuator, Controller)

**Sensors** The system utilizes various types of sensors to collect data about the environment and the trash bin's status, ensuring accurate monitoring and control.

HC-SR04 Ultrasonic Sensor:

* Function: Measures the distance from the trash bin lid to the surface of the waste. This value is used to calculate the fill level of the bin as a percentage. The sensor operates based on the principle of emitting ultrasonic waves and receiving reflected signals, using the time it takes for the wave to bounce back to calculate distance with high accuracy.
* Installation Position: The sensor is fixed beneath the bin lid, pointing downwards towards the waste.
* Key Specifications of HC-SR04:
  + Operating Voltage: 5V DC
  + Static Current: < 2mA
  + Effective Angle: < 15 degrees
  + Measurement Range: 2 cm – 400 cm (some versions may support up to 450 cm)
  + Accuracy: Up to 0.3 cm (depending on environmental conditions and calibration)

YL-83 Rain Sensor Module:

* Function: Detects the presence of rain. The sensor provides a signal (e.g., analog or digital) to the microcontroller, indicating whether it is raining.
* Installation Position: Mounted in a suitable position outside the trash bin, ensuring direct contact with rainwater.
* Key Specifications (of YL-83): (Please add specifications such as Operating Voltage, Output, etc., if available).

**Microcontroller** The ESP32-C6 acts as the central processor for each smart trash bin unit, responsible for coordinating all operations.

* Sensor Connection: The microcontroller is connected to the sensors via GPIO pins. For example:
  + The Trigger pin of HC-SR04 is connected to GPIO 20.
  + The Echo pin of HC-SR04 is connected to GPIO 21 on the ESP32-C6.
  + The signal pin of the rain sensor is connected to an analog or digital GPIO pin (e.g., GPIO 34 for analog or GPIO 23 for digital).
* Embedded Firmware: The software loaded into the ESP32-C6 has the following main responsibilities:
  + Activate the ultrasonic sensor to emit pulses and receive reflected signals.
  + Calculate distance based on the measured time.
  + Read and process data from the rain sensor.
  + Convert distance into the percentage fill level of the trash bin.
  + Control the servo motor to open/close the trash bin lid based on the signal from the rain sensor.
  + Manage Wi-Fi connection and transmit data (fill level, rain status, lid status) to the cloud platform periodically via MQTT protocol in JSON format.
  + Activate the LED warning light system when the waste level reaches predefined thresholds.

**Actuators** Actuators perform physical actions based on data and commands from the microcontroller.

**LED Waste Level Warning System:**

* To provide on-site visual alerts, the system integrates three LEDs into the smart trash bin device.
* These LEDs are connected to different GPIO pins of the ESP32-C6 (e.g., LED 1 connected to GPIO 10, LED 2 to GPIO 11, LED 3 to GPIO 12).
* **Alert Logic:** When the firmware detects that the trash bin's fill level exceeds predefined thresholds, the corresponding GPIO pins will be activated to turn on the LEDs, indicating 3 different alert levels (e.g., green light - low waste, yellow light - nearly full, red light - full).
* This serves as a clear indicator for users or waste collection staff that the bin needs to be emptied.

**SG90 Micro Servo:**

* **Function:** Controls the mechanism for opening/closing the trash bin lid.
* **Connection:** The servo motor is connected to a PWM control GPIO pin of the ESP32-C6 (e.g., GPIO 13).
* **Operating Logic:** When the rain sensor detects rain, the microcontroller will command the servo to close the bin lid. Conversely, when the rain stops, the servo will open the lid back up. This helps protect the waste from getting wet and increases the system's automation.

**Microcontroller**:

* The ESP32-C6 serves as the central processing unit for each smart trash bin.
* The microcontroller is connected to the ultrasonic sensor via GPIO pins. For instance:
* Trigger pin of the HC-SR04 is connected to GPIO 20
* Echo pin is connected to GPIO 21 on the ESP32-C6

The embedded firmware loaded onto the ESP32-C6 is responsible for:

* Triggering the ultrasonic sensor to emit pulses and capturing the reflected echo
* Calculating the distance based on the time measured
* Converting the distance to a fill-level percentage of the bin
* Managing the Wi-Fi connection and transmitting the fill-level data to the cloud platform periodically via the MQTT protocol in JSON payload format
* Activating an LED warning light when the trash reaches a predefined threshold

3.2. Network and Communication

**Network:**

* The smart trash bin device utilizes the integrated Wi-Fi module on the ESP32-C6 to connect to the existing Wi-Fi network at the installation site.
* This allows the device to access the Internet and establish a connection to the cloud platform. The WiFiManager library is implemented in the firmware to simplify and provide flexibility in configuring the Wi-Fi credentials.

**Communication Protocol:**

* The system uses the MQTT (Message Queuing Telemetry Transport) protocol to transmit data from the device to the cloud. MQTT is chosen because it is a lightweight publish/subscribe messaging protocol, ideal for IoT devices that operate with limited resources and require low power consumption.
* Topic: The fill-level data of the trash bin is published to a specific MQTT topic, for example: "khoang\_cach".
* Payload Format: The distance value measured by the ultrasonic sensor (distance\_cm) is converted into a fill-level percentage. Both values are transmitted in a JSON payload format. Node-RED on the cloud platform receives this JSON string, processes it, and transforms the data into a suitable format for storage and visualization.

**Cloud Connectivity:**

* The ESP32-C6 device uses the EspMQTTClient library to establish a connection with an MQTT Broker hosted on a cloud platform (Node-RED).
* Authentication with the MQTT broker is performed using a Client ID (e.g., "TestClient") and the Broker’s IP address (e.g., "192.168.191.92"). The sample code does not use a username/password, which should be noted as a consideration in the system’s security configuration.
* On the Node-RED platform, an MQTT input node is configured to listen for topics from the device. Node-RED then processes the incoming data and stores it into InfluxDB for long-term storage and analysis.
* Notably, when the trash level exceeds the threshold or a critical event occurs, Node-RED triggers a logic flow to send an alert via Gmail. Node-RED supports built-in or external email nodes that allow you to configure a Gmail account and customize the email content (e.g., "The trash bin is full. Please check."). This provides an effective and immediate alerting mechanism for stakeholders.



3.3. Hardware Implementation

During the testing phase, electronic components are arranged on a breadboard for easy connection and verification. The basic circuit design involves connecting the HC-SR04 ultrasonic sensor, rain sensor, servo motor, and waste level indicator LEDs to the ESP32-C6 microcontroller.

The GPIO pins of the ESP32-C6 are used to interface with sensors and control actuators. Specifically:

* The Trigger pin of the ultrasonic sensor is connected to GPIO 20 of the ESP32-C6.
* The Echo pin of the ultrasonic sensor is connected to GPIO 21 of the ESP32-C6.
* The signal pin of the rain sensor is connected to GPIO 22 of the ESP32-C6 (Digital Input/Output pin, or potentially an Analog pin if analog values are used).
* The control pin of the servo motor is connected to GPIO 15 of the ESP32-C6 (PWM-capable pin).
* Three waste level indicator LEDs are connected to separate GPIO pins:
  + The green LED (low/empty waste) is connected to GPIO 12 of the ESP32-C6.
  + The yellow LED (partially full waste) is connected to GPIO 11 of the ESP32-C6.
  + The red LED (full waste) is connected to GPIO 10 of the ESP32-C6.
* A 5V power supply (from an adapter or USB port) powers both the ESP32-C6 and the sensors. For the servo motor, ensure the power supply provides sufficient current for stable operation; a separate power source might be needed if the servo requires more current than the ESP32-C6 can provide to avoid voltage drops.

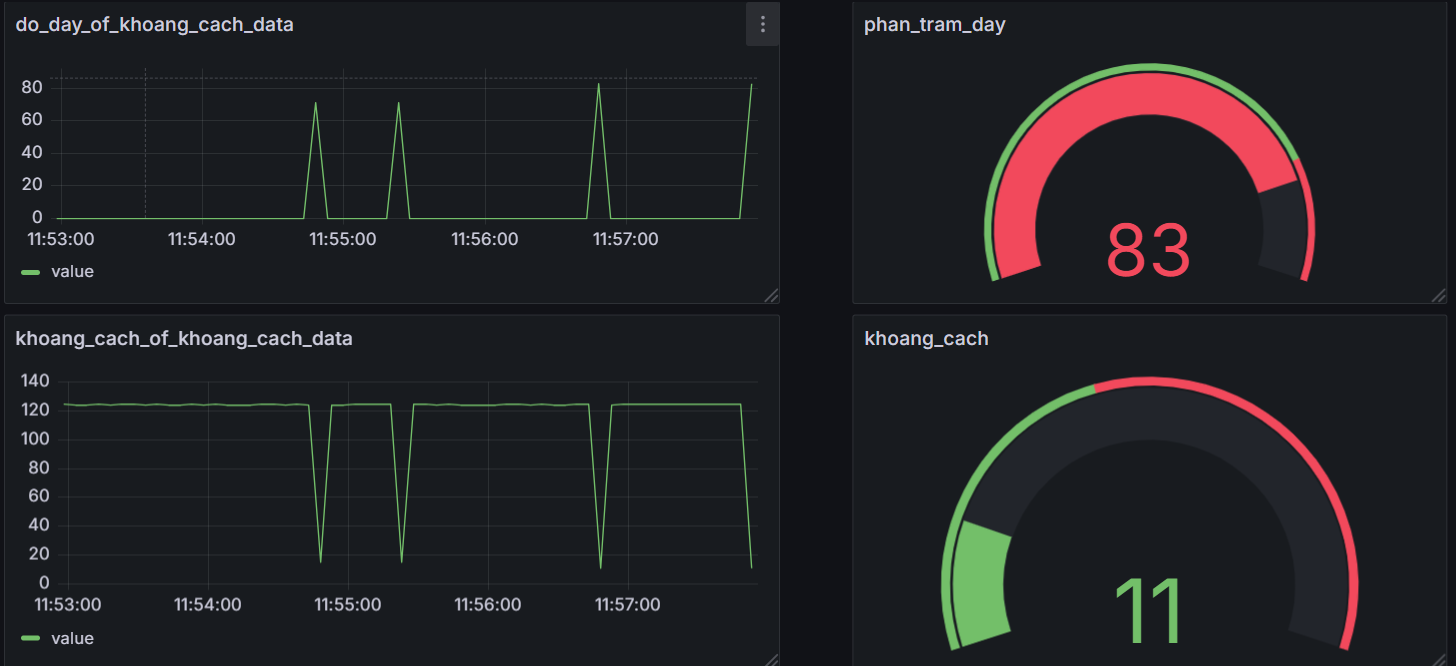
3.4. Data Visualization

**Dashboard Description:** The system utilizes Node-RED Dashboard as the primary interface for data visualization. The dashboard is designed to present information in a clear and real-time manner:

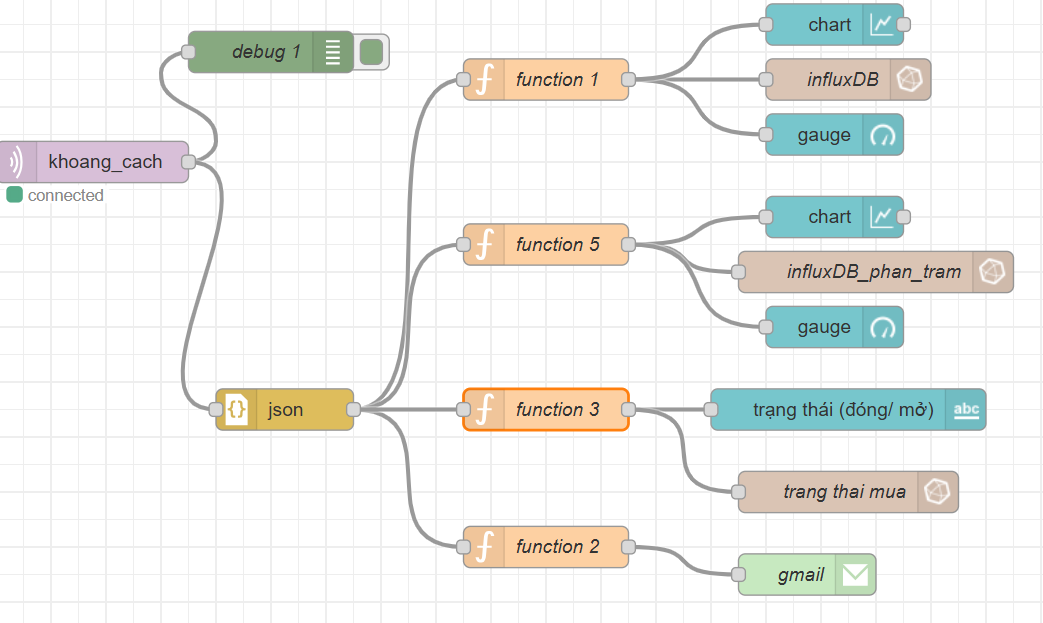
• Numeric display (e.g., Gauge/Single Stat): Shows the measured distance (in cm) and fill level (in percentage) directly from the sensor.  
• Line chart: Visualizes the historical trend of distance and fill level over time, enabling users to monitor and analyze the rise or fall of trash accumulation.  
• Visual alerts: When the measured distance reaches a predefined threshold (i.e., the bin is full), indicators on the dashboard can change color or display alerts to attract user attention.

In addition, Grafana is also configured and used to query data from InfluxDB, offering more advanced data analysis capabilities and the ability to create complex visualizations when needed.

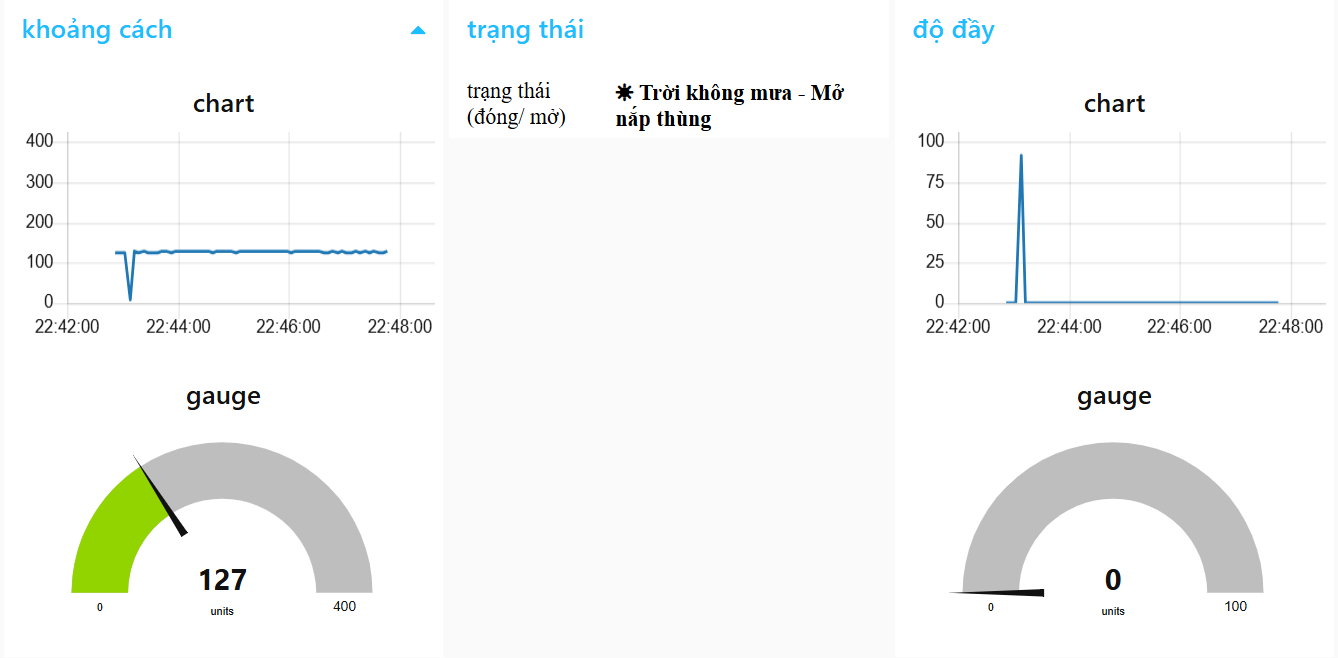
* **Functionality**: The dashboard offers the following key features to support efficient waste management:  
  • Real-time status monitoring: Users can instantly monitor the trash fill level in each bin as the distance data is continuously updated.  
  • Historical data view: Provides the ability to review distance data over various time periods for behavior analysis and prediction.  
  • Notifications: The dashboard displays visual alerts when the distance reaches a predefined threshold (i.e., the bin is nearly full), enabling timely decision-making for waste collection.



**Image Description:** The image above shows the data query and visualization interface in Grafana. The chart displays the distance values (\_value) from the ultrasonic sensor (measuring khoang\_cach), which are stored in the khoang\_cach\_data bucket of InfluxDB. This chart helps monitor the variation in distance over time, thereby indicating the fill level of the trash bin.



This image illustrates the interface of the Node-RED Dashboard, where distance and fill level data are displayed using gauge meters and simple charts, providing a quick overview of the current status of the trash bin.



3.5. Testing and Improvements

The testing process is an essential part of project development, ensuring the system operates reliably and accurately. Below is a description of the test scenarios conducted, the results obtained, as well as the issues identified and the corresponding improvement solutions:

* Sensor Testing:

**Testing Ultrasonic Sensor (HC-SR04):**

* **Accuracy of Distance Readings:** Compare the distance values read by the ultrasonic sensor (displayed on the Serial Monitor and Dashboard) with actual measurements using a tape measure or ruler at various distances (e.g., 10 cm, 50 cm, 100 cm). Record the error and evaluate reliability.
* **Data Stability:** Monitor the sensor's readings over a fixed period to detect noise or abnormal fluctuations. Ensure that the returned values are stable when there is no change in distance.

**Testing Rain Sensor (YL-83 Rain Sensor Module):**

* **Reaction to Water:**
  + Test the sensor's output state when dry (no water) and when wet (drips, mist, pouring water). Verify that the sensor switches states correctly (e.g., from HIGH to LOW or vice versa).
  + Observe notifications on the Serial Monitor or Dashboard to confirm that the rain status is accurately recorded.
* **Sensitivity:** Evaluate the sensor's sensitivity to different amounts of water (e.g., light drizzle, moderate rain, heavy rain) to ensure it reacts appropriately to real-world weather conditions.

**Testing Servo Motor (SG90 Micro Servo):**

* **Lid Open/Close Functionality:**
  + Test the servo's ability to smoothly and fully move the trash bin lid to the closed (ClosedAngle) and open (OpenAngle) positions.
  + Ensure the lid closes tightly when commanded to close and opens completely when commanded to open, without getting stuck or obstructed.
* **Operational Reliability:** Perform multiple consecutive open/close cycles to assess the durability and reliability of the servo motor and lid mechanism.

**Testing LED Warning System:**

* **Reaction to Waste Levels:** Place objects into the trash bin to simulate different fill levels (empty/low, partially full, full) and verify that the corresponding green, yellow, and red LEDs light up correctly according to the programmed logic.
* **Brightness and Visibility:** Ensure the LEDs are bright enough to be clearly visible under various lighting conditions.
* Transmission Testing:
* Wi-Fi and MQTT connection: Ensure that the ESP32-C6 can maintain a stable connection to the Wi-Fi network and the MQTT Broker.
* Data integrity: Verify that the distance data is successfully transmitted to the cloud (Node-RED and InfluxDB) without data loss or packet drops.
* Data transmission frequency: Confirm that data is sent at the configured interval (e.g., every 5 seconds).
* Dashboard Testing:
* Real-time data display: Ensure that the measured distance is accurately displayed and continuously updated on the Node-RED Dashboard (and Grafana, if used).
* LED alert functionality: Check whether the LED turns on/off correctly when the measured distance reaches/exceeds the predefined threshold set in the firmware.
* Gmail alert functionality: Verify that alert emails are successfully sent when the "full" threshold is reached.
* Durability/Energy Testing:
* **Continuous operation evaluation:** Assess the device’s ability to function continuously over a certain period (e.g., several hours) to estimate power consumption and connection stability.
* **Connection stability:** Observe whether frequent disconnections from Wi-Fi or the MQTT Broker occur..

**Test Results**The testing phase yielded positive and encouraging results, confirming the feasibility and stability of the system:

* Sensor accuracy: The ultrasonic sensor performed well in the testing environment, with an average error margin of approximately ±2–5 cm within the 0–200 cm range.
* Data transmission: Data was transmitted reliably to the cloud, with a packet loss rate of less than 1%. The end-to-end latency from device to dashboard typically remained under 2–3 seconds.
* Dashboard functionality: The measured distances were accurately displayed on both Node-RED Dashboard and Grafana. The alert LED worked correctly when the fill level reached the 20 cm threshold. Email notifications were successfully sent within 5–10 seconds of triggering events.
* Durability & energy performance: The system maintained stable operation for 4–6 hours using a 5V power supply. Wi-Fi and MQTT connections were consistently reliable throughout the testing period.

**Identified Issues and Improvement Measures:** Throughout the development and testing process, several challenges were identified and addressed:

Issue 1: Ultrasonic sensor noise

* Problem: Sensor readings occasionally displayed noise due to environmental interference or uneven wave reflections.
* Solution: Implemented a basic smoothing algorithm in the firmware (e.g., averaging 5–10 consecutive readings) to improve data stability and reduce noise.

Issue 2: Temporary Wi-Fi/MQTT disconnections

* Problem: The device experienced occasional brief disconnections from Wi-Fi or the MQTT broker.
* Solution: Integrated an automatic reconnection mechanism using client.loop() in the void loop() function along with the EspMQTTClient library's built-in recovery logic to ensure self-healing connections without manual intervention.

Issue 3: Lack of percentage-based fill indicator

* Problem: The initial dashboard displayed raw distance values in centimeters, making it harder for users to visually interpret how full the trash bin was.
* Solution: *(Planned enhancement)* A feature to convert distance values to fill-level percentages is under development and will be deployed in Node-RED or Grafana. This will provide a more intuitive, user-friendly visual indicator on the dashboard.

Issue 4: Limited testing with a single sensor

* Problem: Current testing was limited to a single sensor, which doesn't reflect real-world deployment scenarios.
* Solution: To evaluate the scalability of the system, future testing will involve multiple sensors/bins to assess the cloud platform’s performance and MQTT protocol handling under larger data loads.

4. Projected Impact

4.1. Accomplishments and Benefits

* **Thành tựu:**
* Successfully built a stable prototype of a smart waste management system.
* Collected real-time data on trash bin fill levels.
* Developed an intuitive user interface for administrators.
* **Lợi ích:**

 **Cost savings:** Reduce unnecessary waste collection trips and optimize routes.

 **Increased efficiency:** Ensure timely waste collection and prevent bin overflow.

 **Environmental improvement:** Reduce odor pollution and the spread of pathogens.

 **Enhanced quality of life:** Create a cleaner and more pleasant urban landscape.

 **Data for analysis:** Provide valuable information for long-term waste management planning.

4.2. Future Improvements

To enhance the effectiveness and expand the applicability of the smart waste management system, several future improvements and developments have been identified:

* **Hardware Improvements:**
* **More durable sensors:** Research and implement more robust sensor types with better resistance to harsh environmental conditions such as high temperatures, high humidity, and dust. This will extend the device's lifespan and reduce maintenance costs.
* **Integration of solar power modules:** Develop solar-powered modules to supply energy to the system, enabling it to operate independently and sustainably, especially in areas with limited access to the power grid.
* **Industrial-grade enclosure design:** Design and manufacture protective enclosures that meet industrial standards—waterproof, dustproof, impact-resistant, and vandal-proof—to ensure device safety and stable operation under all conditions.
* **Software/Cloud Improvements:**
* **AI/Machine Learning integration:** Apply Artificial Intelligence and Machine Learning algorithms to analyze historical data on waste fill levels. AI/ML can more accurately predict when bins will be full, based on collection patterns, waste types, and seasonal or time-of-day variations.
* Development of optimized collection route algorithms: Build or integrate advanced route optimization algorithms that calculate the shortest and most efficient collection paths for garbage trucks, focusing on full bins to save fuel and time.
* Enhanced notification systems: Expand alert channels to include SMS, email, and integration with existing city management platforms to ensure that notifications reach the right personnel quickly and effectively.
* Support for multiple waste types: Upgrade the system to manage and monitor different types of waste (e.g., recyclable, organic, medical waste) by integrating appropriate sensors or using more intelligent data classification.
* **Application Expansion:**
* **Larger-scale deployment:** Scale the system from pilot implementation to larger areas such as entire cities or extensive residential zones, demonstrating the scalability of the solution.
* **Integration with other smart city systems:** Connect and exchange data with other smart urban infrastructure systems (e.g., smart traffic, smart lighting) to create a more connected and efficient urban environment.

5. Team Member Review and Comment

|  |
| --- |
| <ATTACH A TEAM PICTURE HERE> |

|  |  |
| --- | --- |
| NAME | REVIEW and COMMENT |
| **Trần Khắc Nam** | Vai trò của tôi tập trung vào phát triển nền tảng đám mây và giao diện người dùng. Tôi đã nắm vững cách sử dụng các dịch vụ IoT như **Node-RED** để xử lý luồng dữ liệu, **InfluxDB** để lưu trữ dữ liệu thời gian thực và xây dựng **Node-RED Dashboard** (cùng với **Grafana** cho phân tích chuyên sâu hơn) để trực quan hóa dữ liệu. Việc tích hợp dữ liệu thời gian thực, xử lý cảnh báo và gửi thông báo qua **Gmail** là phần thú vị nhất. Dự án này giúp tôi củng cố kiến thức về phát triển web và kiến trúc hệ thống IoT end-to-end. |
| **Lê Văn Chính** | Trong dự án này, tôi chịu trách nhiệm chính về phần cứng và firmware. Thách thức lớn nhất là đảm bảo độ chính xác của cảm biến siêu âm HC-SR04 trong các điều kiện môi trường thử nghiệm và tối ưu hóa việc truyền dữ liệu từ **ESP32-C6** qua MQTT. Tôi đã học được cách làm việc nhóm hiệu quả, kỹ năng debug hệ thống IoT ở cấp độ thiết bị và quản lý tài nguyên vi điều khiển để đảm bảo hoạt động ổn định. |
| **Dương Quang Vinh** | Trong dự án này, tôi chịu trách nhiệm chính về **kiểm thử, tích hợp và tối ưu hóa hệ thống tổng thể**, đảm bảo sự kết nối thông suốt giữa phần cứng, firmware và nền tảng đám mây. Tôi đã phát triển các kịch bản kiểm thử toàn diện để xác minh độ chính xác của dữ liệu cảm biến, độ tin cậy của việc truyền dữ liệu qua MQTT, và khả năng phản hồi của hệ thống điều khiển nắp tự động. Mục tiêu của tôi là đảm bảo rằng mọi thành phần hoạt động hài hòa, mang lại hiệu suất tối ưu và trải nghiệm người dùng ổn định. |

6. Instructor Review and Comment

|  |  |  |
| --- | --- | --- |
| CATEGORY | SCORE | REVIEW and COMMENT |
| IDEA | \_\_/10 |  |
| APPLICATION | \_\_/30 |  |
| RESULT | \_\_/30 |  |
| PROJECT MANAGEMENT | \_\_/10 |  |
| PRESENTATION & REPORT | \_\_/20 |  |
| TOTAL | \_\_/100 |  |