Smart Scale: An IoT-Enabled Weighing Solution

W. P. R. S. Perera, H. A. H. M. Hettiarachchi, K. H. A. J. C. Kumara and S. D. Marasinghe*

Department of Electrical Engineering, Institute of Engineering Technology, Temple Road, Katunayake
*sakunsinghe@gmail.com

Abstract. The Smart Scale: An IoT-Enabled Weighing Solution is an innovative project designed to automate weight measurement, data logging, and remote monitoring through the integration of the Internet of Things (IoT). This system utilizes a load cell sensor to measure the weight of objects and a microcontroller to process and transmit the data to a Google Spreadsheet for real-time storage and analysis. The project offers a user-friendly interface with a simple push-button control and displays the weight data on an LCD screen for immediate viewing. Traditional weighing methods often involve manual data entry, which can lead to errors, inefficiencies, and challenges in accessing historical data. This IoT-enabled smart scale system addresses these issues by providing accurate and automated weight measurement, along with the ability to monitor and analyze data remotely. The integration with cloud-based tools, such as Google Sheets, ensures that data can be accessed and analyzed from anywhere, offering flexibility and convenience for users. This project prioritizes simplicity, portability, and real-time functionality, making it an ideal solution for small-scale producers, such as in tea production or honey harvesting, where precise weight measurement is critical.

Keywords: Smart Scale, IoT Weighting System, Real-Time Monitoring, Cloud Data Storage, Local Server

1 Introduction

The Internet of Things (IoT) has transformed many industries by enabling smarter, automated, and more efficient solutions. One significant innovation is the **Smart Scale: An IoT-Enabled Weighing Solution,** which modernizes traditional weighing methods by introducing automation, remote monitoring, and real-time data analysis. This system provides an accurate and efficient alternative to manual weighing, which is often slow, laborintensive, and prone to human error. By leveraging IoT technology, the Smart Scale reduces inaccuracies and streamlines the weighing process, improving productivity and reliability in various operational environments [1] [2]. This project utilizes a **load cell sensor**, a device that converts weight into an electrical signal, paired with the HX711 amplifier module to precisely measure the weight of objects. The data is collected and processed by a microcontroller such as the ESP8266 or ESP32, which plays a central role in handling sensor inputs, performing necessary calibrations, and managing data transmission. The system offers flexibility by sending weight data to either a cloud-based platform like Google Sheets or a local server environment running XAMPP (Apache, MySQL, PHP). This dual option supports both internet-enabled and offline scenarios, ensuring continuous data logging and accessibility. Additionally, a local LCD screen displays real-time weight readings, providing users with immediate feedback, while remote web interfaces enable monitoring, historical data analysis, and reporting through intuitive dashboards and alerts [3] [4] [5].

Accurate and timely weight measurement is critical in industries such as tea production**,** honey harvesting**,** agriculture, food processing, pharmaceutical manufacturing, and other small to medium-scale enterprises. These sectors often rely on manual weighing processes that are time-consuming and susceptible to mistakes, lacking systematic data storage for performance analysis and traceability. The Smart Scale addresses these pain points by integrating IoT-enabled automation with affordable and readily available hardware components. This integration not only improves accuracy and efficiency but also supports data-driven decision-making by maintaining comprehensive historical records accessible from anywhere at any time, enhancing transparency and operational control [6].

The Smart Scale system is designed with portability and ease of use in mind. Its compact hardware footprint and low power consumption make it suitable for deployment in diverse environments, including remote or outdoor locations where traditional weighing infrastructure is impractical. The system's modular architecture allows for easy customization and scalability, enabling users to adapt it to various weighing requirements and integrate additional sensors or communication modules as needed. The primary goal of this project is to develop a portable, user-friendly, and cost-effective IoT weighing system that enhances operational efficiency. By automating data collection and storage, the system reduces human intervention, minimizes errors, and provides businesses with easy access to real-time and historical weight data. This capability empowers small-scale industries to optimize resource management, improve quality control, and scale their operations with reliable technological support.

While several existing solutions demonstrate IoT-based weighing scales, this project differentiates itself through several key innovations. Firstly, it offers a hybrid data logging architecture, enabling seamless transmission to both cloud platforms (Google Sheets) and local servers (XAMPP), ensuring operational resilience in environments with intermittent internet connectivity. Secondly, the system is designed as a fully portable, self-powered unit, integrating a battery management system, which is often omitted in comparable designs. Finally, this work provides a complete end-to-end solution from hardware sensor interfacing and calibration to a sophisticated web-based dashboard for data visualization and management, offering a level of practicality and user accessibility that extends beyond the scope of typical prototype demonstrations.

2 Methodology

The methodology for this project involves a systematic approach to designing, implementing, and testing an IoT-enabled weighing system. The development of the Smart Scale follows the steps of requirement analysis, design, component selection, and integration, followed by system implementation and testing. The methodology encompasses both hardware and software aspects, where the hardware includes a load cell, HX711 amplifier module, ESP32 microcontroller, and cloud storage for data, while the software is focused on controlling the system, collecting data, and visualizing it in real-time. The IoT-enabled weighing system operates by measuring the weight of items through the load cell, which transduces weight into an electrical signal.

This signal is amplified by the HX711 module, and the resulting data is then processed by the ESP32 microcontroller. The microcontroller sends the data to a cloud platform for real-time monitoring and analysis. Additionally, an LCD screen is used to display real-time weight readings, providing a direct interface for users to interact with the system. The system is designed to be user-friendly and cost-effective, ensuring that it can be easily deployed in various small-scale industrial settings.

2.1 System Design

The system is designed to perform real-time weight measurement using a load cell connected to an HX711 amplifier module, which in turn is connected to an ESP32 microcontroller. The ESP32 processes the analog data, converts it into readable weight units, and displays the value on an LCD. Once stabilized, the data is sent via Wi-Fi to a Google Sheet for cloud storage. The control flow begins with powering on the device, initializing modules, calibrating the sensor if required, continuously reading the weight, checking for stability, and finally displaying and transmitting the value. Push buttons allow user interaction for reset or calibration (Fig. 1).

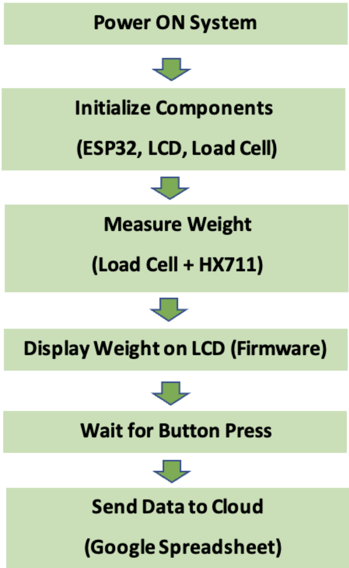


Fig. 1. System Design Flow Chart

2.2 Materials and Components

The Smart Scale project integrates several key hardware and software components to build a functional IoT-based weighing system. Below is a list of hardware components and software tools shown in Tables 1 and Table 2.

Hardware Components

Table 1. Components with Specification

Component	Function/Purpose
ESP32 Microcontroller	Processes sensor data, handles Wi-Fi communication, and sends data to the server/cloud.
Load Cell with Structure (200 kg)	Measures object weight and converts it into analog signals.
HX711 Amplifier Module	Amplifies and converts analog load cell signals to digital for microcontroller input.

TP4056 Charging Module	Manages safe charging of the lithium-ion battery.
20x4 LCD Display	Displays weight values and system messages in real time.
I2C Interface Board	Simplifies LCD communication with ESP32 via I2C protocol.
Lithium-ion Battery	Supplies portable power to the system.
Push buttons (stainless steel)	Used for user input: Send & Tare (Reset).
Power Boost Voltage Converter	Converts lower battery voltage to a stable 5V required for ESP32 and peripherals.
Jumper Wires (Set)	Connects all components in the circuit.
Mini Rocker Switch	Turns the system on or off manually.
Enclosure/Packaging Box	Protects components and ensures safe, neat installation.

Software Tools

Table 2. Tools with Specifications

Software Component	Function/Purpose
Arduino IDE	Used to write, compile, and upload code to the ESP32 microcontroller.
PHP (on XAMPP)	Processes incoming weight data and stores it in the MySQL database.
MySQL (on XAMPP)	Stores real-time and historical weight data for retrieval and analysis.
phpMyAdmin	Provides a web interface to manage the MySQL database.

2.3 Circuit Assembly

The load cell is connected to the HX711 amplifier module, which amplifies the signal from the load cell and converts it into a digital output. Then, the HX711 is connected to the ESP32 microcontroller via the appropriate pins (SCK and DT for data transfer). The ESP32 microcontroller is connected to the HX711, LCD, push button, and TP4056 charging module. The TP4056 charging module is used to charge a lithium-ion battery, which powers the ESP32 and other components, allowing for portability. The I2C pins (SDA, SCL) are connected between the ESP32 and the LCD module. The push button is wired to an input pin of the ESP32. Fig. 2 shows the complete circuit.

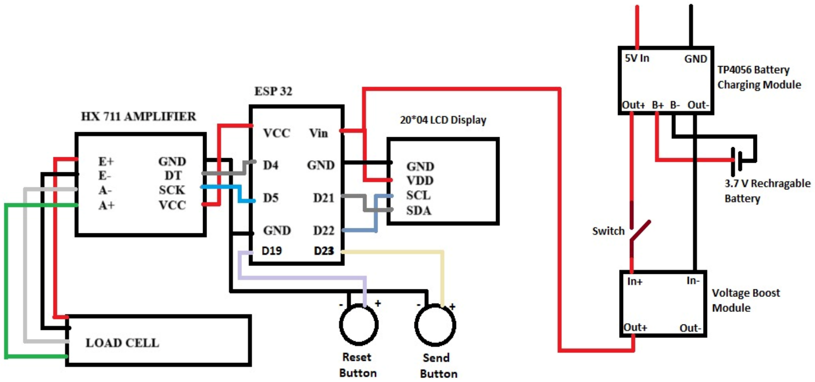


Fig. 2. Circuit Diagram

2.4 Software Development

The ESP32 was programmed to initialize the sensor, calibrate it using known weights, and convert the digital readings into human-readable weight measurements. The microcontroller connected to a Wi-Fi network and periodically transmitted weight data to a web server via HTTP requests. The firmware was written using the Arduino IDE and involved the use of libraries for sensor communication and HTTP client handling. The data transmitted included the weight value, and on the server side, the date and time of each reading were also recorded.

Software Implementation Details

To ensure reproducibility, the key aspects of the software implementation are detailed below. The firmware was developed using the Arduino IDE (v2.3.2) and relied on the following libraries: HX711_ADC (v2.0.1) for sensor communication, LiquidCrystal_I2C (v1.1.4) for the LCD, and the built-in WiFi and HTTPClient libraries for connectivity.

The scale must be calibrated against known weights; this process calculates a calibration factor which is stored in non-volatile memory. The main loop continuously reads the load cell value, converts the raw sensor data to grams using the stored factor, and displays it on the LCD. The conversion formula is:

weight_grams = (raw_value - offset) / calibration_factor; // Pseudo-code

When the user presses the 'Send' button, the firmware checks if the reading is stable. If stable, it constructs an HTTP POST request to the server. The request sends the weight value as a query parameter:

POST /post-data.php?weight=5076.72 HTTP/1.1

On the server side, a PHP script (post-data.php) receives this request. It validates the input, sanitizes the data to prevent SQL injection, generates a timestamp, and inserts the record into the MySQL database using a simple SQL query:

INSERT INTO scale_data (weight, date, time) VALUES (5076.72, '2025-06-13', '14:30:05')

2.5 Local Server Configuration Using XAMPP

Server Environment Setup

XAMPP software was used to configure the local server environment. It provided an Apache web server to host PHP scripts and a MySQL server to manage the database. Both modules were launched and controlled through the XAMPP control panel shown in Fig. 3.

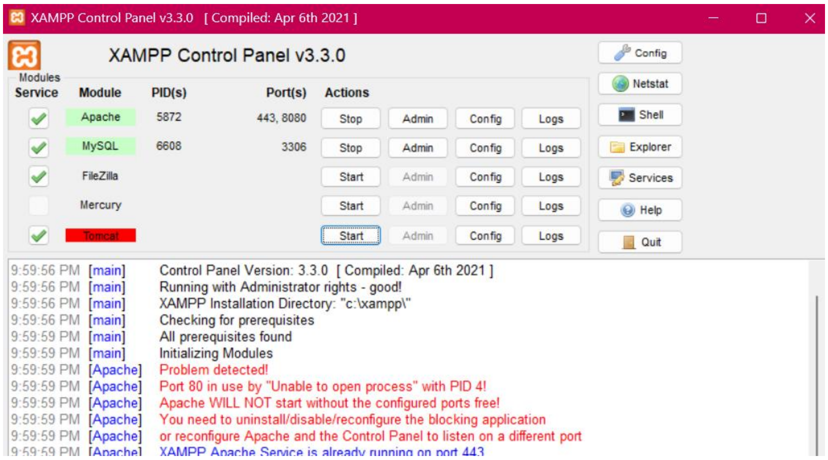


Fig. 3. XAMPP Control Panel with Apache and MySQL Services Running

Database Creation and Table Structure

A MySQL database named smartscale1 was created using phpMyAdmin. Within it, a table named "scale data" was designed to store the following fields shown in fig. 4:

- ID: Auto-increment primary key
- Weight: Float value representing measured weight
- Date: Date of the reading
- Time: Time of the reading

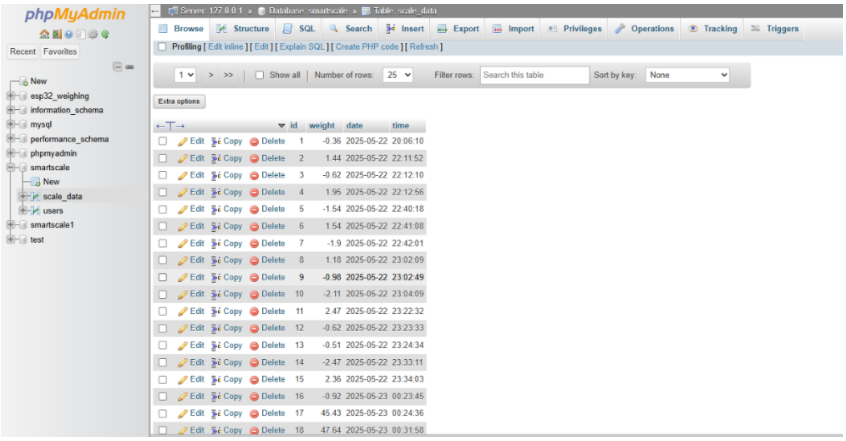


Fig. 4. Database Table Structure in phpMyAdmin

PHP-Based Data Handling

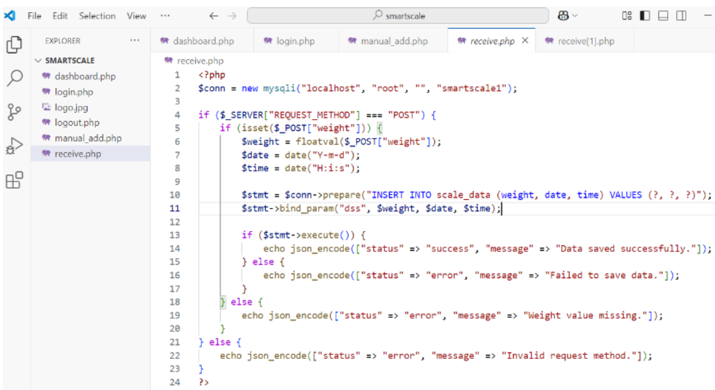


Fig. 5. PHP Script for Data Insertion

A PHP script shown in Fig. 5 was developed to receive HTTP POST requests sent by the ESP32. This script parsed the received weight data, generated a timestamp using the server clock, and inserted the data into the MySQL table.

3 Testing and Calibration

Testing and calibration are essential phases to ensure the proper functioning, reliability, and accuracy of the Smart Scale system. These processes help identify issues, validate system performance, and determine how well the project meets its objectives.

3.1 Initial Testing

After the hardware and software are integrated, the system is tested to ensure that the components work together seamlessly. During testing, the system is checked for weight accuracy, LCD functionality, button press response, and cloud connectivity.

3.2 Calibration of Load Cell

Calibration was performed using a set of known weights shown in Fig. 6 to determine the sensor's response factor. The calibration factor was adjusted by comparing sensor readings with actual weight values until high accuracy was achieved.

Testing included:

- Taring the scale to zero before use.
- Measuring sample weights from 100 g to 5 kg.
- Verifying the consistency of results over time.

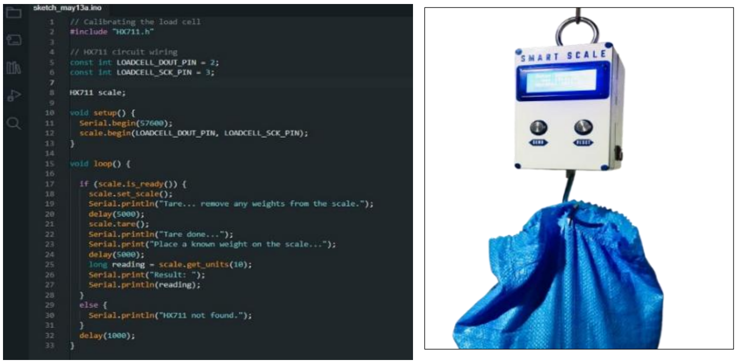


Fig. 6. Calibration Procedure with Known Weights

3.3 Connect to local Wi-Fi.

The microcontroller (such as ESP8266 or NodeMCU) is configured to connect to a local Wi-Fi network using a predefined SSID and password shown in Fig. 7. Upon successful connection, the system retrieves the current date and time from an online time server (via NTP) and displays it on the LCD screen along with the connection status. This ensures proper time-stamped data logging before sending weight data to the cloud.

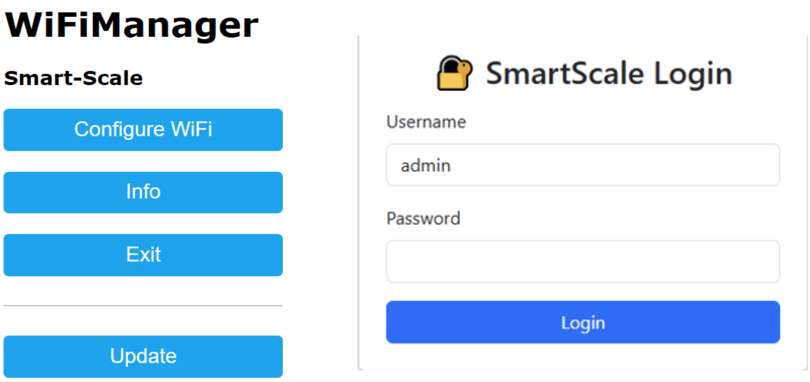


Fig. 7. Wi-Fi manager with login interface

3.4 Results

The final prototype was tested in a real-world scenario. The prototype device is shown in Fig. 8. The system was powered continuously and validated over several hours to ensure performance. The local database captured the weight values accurately and in real-time, with successful transmission of data from sensor to server.

The measurements from the two scales (Fig. 8: real-world scale; Fig. 9: smart scale) show a slight discrepancy. The real-world scale recorded a weight of 5070g, while the smart scale displayed 5076.72g, resulting in a +6.72g error (0.13% deviation). This minor difference is likely due to calibration drift, environmental factors, or variations in placement on the smart scale. Given the real-world scale's declared tolerance of average +4.9 g.

The Google spreadsheet documents shown in Fig. 11 record the weight measurements captured by the weighing system in real time. The sheet consists of three columns which are Date, Time, and Weight (in grams), which together provide a clear, timestamped record of all weighing performed by the load cell and HX711 sensor.

Fig. 8. Final Smart Scale Product

Fig. 9. 5 kg Rice bag measurement using scale

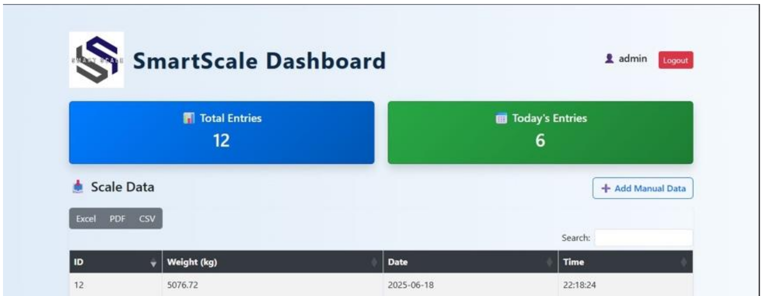
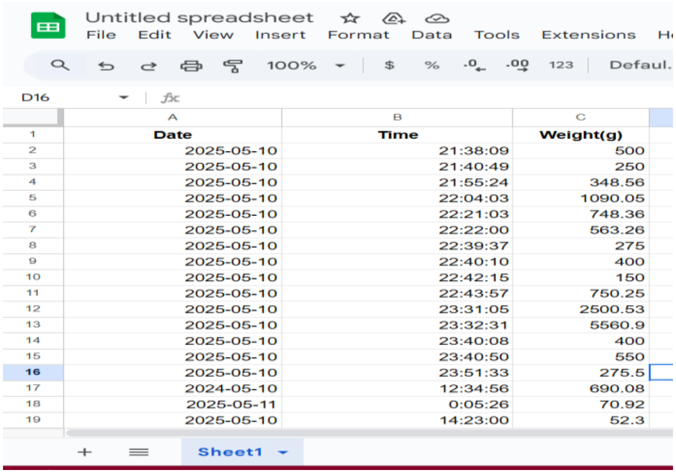


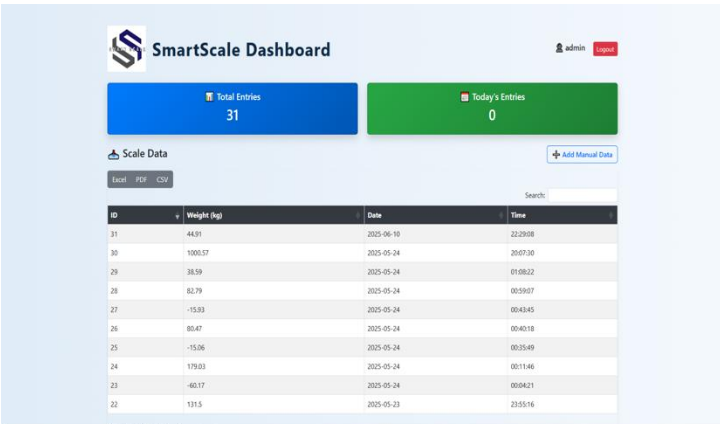
Fig. 10. 5kg Rice bag measurement using Smart Scale

This data is valuable for a range of purposes, including keeping a permanent backup of measurements, analyzing trends or fluctuations in weight over time and generating reports or charts for further insight. Furthermore, because the data is stored on Google Sheets, it can be accessed remotely from any location with an internet connection, allowing for convenient and flexible monitoring of the weighing process and this approach efficiently integrates real-time data collection with permanent storage and analysis, making it a useful tool for small-scale industrial applications and inventory control.



	A	B	C
1	Date	Time	Weight(g)
2	2025-05-10	21:38:09	500
3	2025-05-10	21:40:49	250
4	2025-05-10	21:55:24	348.56
5	2025-05-10	22:04:03	1090.05
6	2025-05-10	22:21:03	748.36
7	2025-05-10	22:22:00	563.26
8	2025-05-10	22:39:37	275
9	2025-05-10	22:40:10	400
10	2025-05-10	22:42:15	150
11	2025-05-10	22:43:57	750.25
12	2025-05-10	23:31:05	2500.53
13	2025-05-10	23:32:31	5560.9
14	2025-05-10	23:40:08	400
15	2025-05-10	23:40:50	550
16	2025-05-10	23:51:33	275.5
17	2024-05-10	12:34:56	690.08
18	2025-05-11	0:05:26	70.92
19	2025-05-10	14:23:00	52.3

Fig. 11. Google Sheet Update Data



ID	Weight (kg)	Date	Time
31	44.91	2025-06-10	22:29:08
30	1000.57	2025-05-24	20:07:30
29	38.59	2025-05-24	01:08:22
28	82.79	2025-05-24	00:59:07
27	-15.93	2025-05-24	00:43:45
26	80.47	2025-05-24	00:40:18
25	-15.06	2025-05-24	00:35:49
24	179.03	2025-05-24	00:11:46
23	-60.17	2025-05-24	00:04:21
22	131.5	2025-05-23	23:55:16

Fig. 12. Web Interface Dashboard

The "Smart Scale Dashboard", a web-based interface designed for the weighing system. The dashboard shown in Fig. 10 and Fig. 12 provides a clear and comprehensive view of all the weight measurements collected by the system. At the top, it shows total entries representing all the recordings made so far. It also highlights "Today's Entries", indicating new recordings for the day. The main table below lists all the recorded weights alongside their IDs, dates, and times of measurement. Each row corresponds to a weighing event, and the table makes it easy to search, filter, and view historical data. The interface also includes convenient options to export the data to Excel, PDF, or CSV, and an "Add Manual Data" button to manually insert a record if needed and this dashboard plays a key role in the project by offering a centralized platform to monitor, manage, and analyze weighing data in real time, making it a valuable tool for small-scale industrial applications and inventory control.

Table 3. Summary Table

Item	**Reference Weight (g)**	**Smart Scale Weight (g)**	**Absolute Error (g)**
Sugar	502g	506.23g	+ 4.23g
Salt	1010g	1014.63g	+4.63g
Flour	3006g	3010.29g	+4.29g
Rice	5070g	5076.72g	+6.72g
A Stone	827g	832.54g	+5.54g
Average			+4.94 g

The performance of the Smart Scale was quantitatively evaluated against a set of reference weights. As summarized in Table 3, the system demonstrated a high degree of accuracy. The mean absolute error (MAE) across all test items was +4.94 grams.

3.5 Comparative Analysis

Table 4. Features and Cost Comparison

Feature	**Proposed Smart Scale**	**Commercial Digital Scale**
Total Cost	Rs.7,790	Rs. 19,541
Cloud Connectivity	Yes (Google Sheets)	No
Locale Server Logging	Yes (XAMPP)	No
Web Dashboard	Yes (Custom PHP/MySQL)	No
Portability(Battery Power)	Yes	Yes
User Interface	LCD + Web Dashboard	LCD
User Calibration	Yes	Often restricted
Declared Accuracy	<1%	Often <0.1%

The proposed Smart Scale system delivers a unique value proposition, strategically balancing cost, functionality, and accessibility. As illustrated in Table 4, the system occupies a distinct niche by offering a combination of local and cloud data flexibility, user repairability, and portability at a fraction of the cost of commercial digital scales. This makes it an ideal and practical solution for small-scale, budget-conscious applications, such as agricultural co-operatives, artisanal production, and inventory management.

4 Discussion

The Smart Scale project successfully addressed the primary problems outlined at the outset, such as the lack of automation in weight measurement, limited portability of traditional scales and the absence of digital recordkeeping. The addition of IoT functionality proved to be a transformative element, enhancing the system's capabilities by enabling remote access, real-time data monitoring, and cloud-based storage. These upgrades offered several advantages over traditional, manual scales. Notably, the ability to track weights digitally eliminates the need for paper-based logs and offers users the convenience of accessing historical data at any time. The integration of cloud technologies enables seamless remote monitoring, which can improve operational efficiency. The system performed well during field tests, demonstrating its potential for scalability and adaptability across various industries, from agriculture to small manufacturing sectors. Although the system remains in its early stages, the results suggest that it holds great promise for widespread adoption, particularly in industries where accurate weight tracking is essential.

5 Conclusions

The Smart Scale project achieved its objectives by successfully developing a compact, reliable, and user-friendly weighing system with integrated IoT capabilities. By combining hardware and software components, the system was able to provide real-time weight tracking, digital logging, and cloud-based data storage. The system demonstrated its practical applicability, especially in industries like agriculture, tea and honey production, and small-scale manufacturing, where accurate and portable weighing solutions are essential. The project also contributes to the advancement of digital transformation in these industries by introducing IoT-enabled devices that improve operational efficiency and data management. While the system is still in its early stages, it represents a significant step toward the development of more intelligent, automated, and accessible weighing solutions.

Reference

1. A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, Internet of Things for Smart Cities, IEEE Internet of Things Journal, 1(1), pp. 22–32, Feb. 2018, Available: https://ieeexplore.ieee.org/document/6740844
2. M. Alam, IoT Weighing Scale with HX711 Load Cell & ESP8266, How To Electronics, Feb. 08, 2020. https://how2electronics.com/iot-weighing-scale-hx711-load-cell-esp8266/#google_vignette
3. A. Parajuli, Insert Data into MySQL Database with ESP8266 Development Board, IoT Projects Ideas, Jun. 20, 2020. https://iotprojectsideas.com/insert-data-into-mysql-database-with-esp8266/
4. Data Logging with PHP + MySQL + ESP8266, Arduino Project Hub, 2022. https://projecthub.arduino.cc/yilmazyurdakul/data-logging-with-php-mysql-esp8266-a33ab0
5. Log Data into MySQL Database using NodeMCU Development Board, Circuitdigest.com, 2021. https://circuitdigest.com/comment/35081(accessed Jun. 13, 2025).
6. M. S. Munna and T. C. Mallick, Transforming weight measurement: a cutting-edge IoT-enabled smart weight machine for centralized price control of products, International Journal of Scientific Reports, 10(8), pp. 269–274, Jul. 2024, doi: https://doi.org/10.18203/issn.2454-2156.intjscirep20241991.