RSET
RAJAGIRI SCHOOL OF
ENGINEERING & TECHNOLOGY
(AUTONOMOUS)

Mini Project Report On

Smart Billing Cart

Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology

in

Electronics and Communication Engineeering

By

Maria Siju Joseph (U2201126)
Keziah Mariam Varughese (U2201117)
K Vijay Krishnan (U2201115)
Milan Thomas C (U2201133)

Under the guidance of

Ms. Soni Monica

Department of Electronics and Communication Engineering
Rajagiri School of Engineering & Technology (Autonomous)
(Parent University: APJ Abdul Kalam Technological University)
Rajagiri Valley, Kakkanad, Kochi, 682039
April 2025

CERTIFICATE

This is to certify that the mini project report entitled ***Smart Billing Cart*** *is a bonafide record of the work done by* ***Maria Siju Joseph (U2201126)*** *, submitted to Rajagiri School of Engineering & Technology (RSET) (Autonomous) in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology (B. Tech.) in Electronics and Communication Engineering during the academic year 2024-2025.*

Project Guide	Project Coordinator	HoD
Ms. Soni Monica	Ms. Ramitha Rajesh	Dr.Jisa David
Assistant Professor	Assistant Professor	Associate Professor
Dept. of ECE	Dept. of ECE	Dept. of ECE
RSET	RSET	RSET

ACKNOWLEDGEMENT

I wish to express my sincere gratitude towards **Rev. Fr. Dr. Jaison Paul Mulerikkal(CMI)**, Principal of RSET, and **Dr. Jisa David**, Head of the Department of Electronics and Communication Engineering for providing me with the opportunity to undertake my project, Smart Billing Cart.

I am highly indebted to my project coordinators, **Ms. Ramitha Rajesh** and **Ms. Ameera Sathar** for their valuable support.

It is indeed my pleasure and a moment of satisfaction for me to express my sincere gratitude to my project guide **Ms. Soni Monica** for her patience and all the priceless advice and wisdom she has shared with me.

Last but not the least, I would like to express my sincere gratitude towards all other teachers and friends for their continuous support and constructive ideas.

Abstract

The project introduces a smart billing cart that increases traditional shopping and checkout experience through automation, accuracy and security. The system uses an ESP32 microcontroller, EM-18 RFID reader, HX711 load cell and 16x2 LCD display, which are all programs through Arduino Ide. When a product is scanned using RFID reader, the system motivates the user to weigh it; PHP-Managed database has product descriptions such as name, cost and weight. Only when the measured weight matches the stored weight, the item is confirmed and added to the virtual cart. Re-scanning the RFID tag removes the item after uniform verification. A custom developed in Android studio handles the Android app billing and payment - the checkout button only activates when all weight values are accurate, leading to a safe QR code payment interface.

Compared to existing smart carts that often depend on barcode scanning or RFID with limited verification, the system introduces double-latter verification using both RFID and weight sensors, which reduces billing errors and theft. Traditional systems either lack real-time recognition or post-checkout depends on the safety check, while this solution ensures that all are made at the point of selection. Additionally, the integration of a mobile app for payment increases the user's convenience and streamlines the entire purchase process, making it rapid, more reliable and difficult to manipulate..

Contents

List of Abbreviations

- ESP32 - Espressif Systems Protocol 32-bit Microcontroller
- RFID - Radio Frequency Identification
- PHP - Hypertext Preprocessor
- QR - Quick Response
- App - Application
- IoT - Internet of Things
- IP - Internet Protocol

List of Figures

List of Tables

Chapter 1

Introduction

Traditional billing systems in retail stores often lead long queues, manual errors and potential theft. To overcome these issues, the project introduces a smart billing cart that automatically explore and billing using ESP 32, EM -18 RFID reader, HX 711 load cell and 16X2 LCD screen. Products are scanned via RFID, and their weight is verified using a load cell against PHP-based database. A custom manages the Android app billing and enables checkouts only when all items are verified, leading to a safe QR code payment system. This solution improves speed, accuracy and safety in the purchase process.

1.1 Background

Conventional checkout procedures at stores are manual where consumers stand in long lines to have every item scanned and charged by a cashier. It is time-consuming, prone to errors, and not efficient, particularly during rush hours. As consumer behavior shifts, demand for automated checkout systems that accelerate the process of checkout, minimize human error, and enhance the customer's experience increases.

Smart billing carts have been brought in as an alternative, and automated item detection and billing are features of it. But most current systems have RFID or barcode scanning separately without any verification step to confirm the scanned item is indeed present within the cart. This leaves scope for fraudulent practice or accidental misbilling.

The Smart Billing Cart used here addresses such challenges by combining RFID product scanning (with EM-18 reader) with dynamic weight verification using HX711 load cell. After RFID scanning a product, the product is asked to be weighed and then matched if it is in line with the stored value in the database. The item is placed to the bill or subtracted when successfully verified, in case scanned multiple times. This two-layer authentication provides greater accuracy and eliminates the possibility of theft or billing errors.

A tailor-made Android application, based on Android Studio, is responsible for the bill process and communicates with the shopping cart through Wi-Fi. It only allows for the checkout choice once all products' weights have been confirmed in the correct manner, resulting in a QR code payment system. The project provides an efficient, secure, and convenient alternative option to current bills in retail systems.

1.2 Problem Definition

Traditional billing systems are time-consuming and prone to errors or theft due to lack of item verification. There is a need for an automated solution that combines product identification and weight validation to ensure accurate and secure billing.

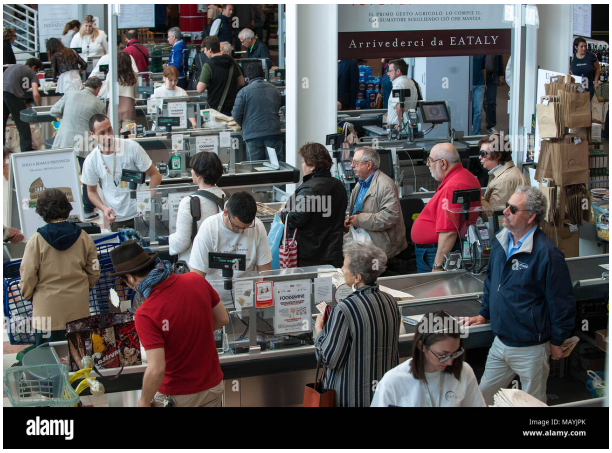


Figure 1.1: Shopping mall during peak hours

1.3 Scope and Motivation

The project scope in this case is to create and develop a Smart Billing Cart that simplifies shopping by automating product recognition, weight checking, and billing. The system employs an ESP32 microcontroller, EM-18 RFID reader, HX711 load cell, and a 16x2 LCD display with programming via the Arduino IDE. A PHP backend manages product data storage, and a proprietary Android app manages billing and payment. The cart checks items in real time by comparing RFID tags scanned to actual item weights, allowing checkout only when everything checks out. This solution scales for supermarket and retail use and can be integrated further with inventory and analytics systems.

The idea for this project comes from the problems we still see in today's retail checkout systems. Manual billing takes a lot of time and often leads to mistakes, which can frustrate customers and cause losses for store owners. While there are smart carts out there, many of them just scan items without actually checking if the product is really in the cart, which leaves room for errors or even misuse. Our project aims to fix that by adding an extra step—checking the product's weight against a database to make sure everything matches before allowing checkout. By combining this with a simple mobile app for billing and payment, we're creating a faster, safer, and more reliable way to shop.

Table 1.1: Components and its functions

Sl No	**Component**	**Function**
1	EM-18 Reader	Scans RFID tags of products
2	HX711 Load Cell	Measures item weight for verification
3	ESP32	Controls hardware and communicates with app
4	PHP + Database	Stores product info (name, weight, price)
5	Android App	Displays bill, enables secure payment

1.4 Objectives

1) **To design a smart billing cart** with the help of ESP32, EM-18 RFID reader, HX711 load cell, and a 16x2 LCD display to simplify the process of product scanning and billing.

2) **To incorporate real-time item authentication** through comparison of every scanned item's weight with the stored value in a PHP database for more accurate billing.

3)**To develop a user-friendly Android application** using Android Studio that would allow users to view their cart, manage items, and pay securely through a QR code.

4)**To automatically remove items** by letting the user rescan RFID tags and verify the weight difference in order to update the bill.

5)**To integrate a secure checkout mechanism** that only enables payment when all the items in the cart are correctly validated so that frauds and errors cannot occur.

6)**For enhancing the shopping experience** by shortening checkout time, avoiding long lines, and increasing security within retail settings.

1.5 Challenges

One of the main challenges was getting the weight sensor to match the database values accurately— even small differences could cause problems during validation. Making sure the ESP32 stayed connected and synced smoothly with the Android app was also important to keep everything running in real-time. On top of that, we had to design the app in a way that only lets users check out when every item is properly verified, which meant carefully connecting all the hardware and software parts to work together perfectly.

1.6 Assumptions

1)Each product has a unique RFID tag and a predefined weight stored in the database.
2)A stable Wi-Fi connection is available for communication
3)Users will scan and weigh one item at a time for accurate verification.
4)The weight sensor is properly calibrated and placed on a stable surface.
Users will follow proper scanning and checkout instructions through the app

1.7 Societal / Industrial Relevance

Societal Relevance

- Improved Shopping Experience: The Smart Billing Cart helps customers avoid long checkout lines by automating the billing process, making shopping faster and more convenient.
- Increased Billing Transparency: By displaying product details like name, price, and weight through the app and LCD screen, customers can clearly see what they're being charged for.
- In a post-pandemic world, this system supports a safer, more hygienic shopping experience by minimizing human contact during checkout.

Industrial Relevance

- Retail Stores and Supermarkets: The Smart Billing Cart can automate the checkout process, reduce manpower requirements, and minimize billing errors, leading to faster customer service and improved operational efficiency.
- Franchise Retail Chains: With centralized databases and connected mobile apps, the system can be scaled and used across multiple store locations, ensuring uniform billing and data tracking.
- Self-Service Kiosks: The technology can be adapted into self-checkout stations where users can scan and verify items on their own, reducing the need for traditional cashiers.

1.8 Organization of the Report

This report is structured into six main chapters, starting with Chapter 1: Introduction, which outlines the project background, motivation, objectives, and relevance. Chapter 2: Literature Survey presents a review of existing technologies, with a comparison between EM-18 and RC522 RFID modules and identification of gaps. Chapter 3: Methodology explains the problem statement, system design using block and circuit diagrams, and a brief on cost estimation. Chapter 4: Embodiment and Detailed Design covers the working principle, detailed design at the component level, and software tools utilized. Chapter 5: Results and Discussions showcases the final prototype, includes demo photos, and discusses key challenges and task distribution. Chapter 6: Conclusion and Future Scope summarizes the outcomes and suggests potential improvements. The report concludes with references, publication details, and appendices for presentations and curriculum mapping.

Chapter 2

Literature Survey

2.1 Introduction

As retail technology rapidly changes, the standard checkout experience is being supplanted by technology that offers faster or greater ease of transactions. One of the newest options that seems to have great promise is automatic billing carts that use RFID (Radio Frequency Identification) and weight sensors. These sensors make it easy to keep track of products added to the cart and vouch for a security purchase without the manual scanning process at checkout. In recent years, RFID has emerged as a leading technology in reducing friction within retail. RFID tags are placed in products that are detectable by RFID readers without line-of-sight, unlike barcode scanners. RFID provides automatic identification of items being placed into shopping carts. RFID systems reduce human error, increase speed, and enhance the customer experience by reporting totals, in real-time, which then initiates the payment process.

Weight sensors are also embedded within these systems to enhance accuracy and security. The weight sensors are intentionally located in the cart to detect the weight of the products and compare that information to the value from the RFID tags. If the item weight does not match what is expected, the system can inform the user to review their selection.

A study by Prof. M.T. Dangat et al. (2025) studied the implementation of weight sensors and RFID systems to avoid user error, such as placing the wrong item or missing an item in the cart. The authors noted that the implementation of RFID and weight sensors would provide a more secure and accurate transaction. RFID and weight sensors work together to ensure items placed in the cart are accurately billed, as well as theft or fraud prevention. Most automatic billing cart systems have three main components: an RFID reader, weight sensors, and a processing unit. The RFID reader obtains data

from RFID tags affixed to the products, and the weight sensors measure the weight of the goods in the cart. Then, the processing unit combines data from both sources, computes the total cost, and readies the system for payment.

Similarly, Amazon's Just Walk Out process serves as an example of an actual system used in retail that utilizes RFID and weight sensors. Using this system, items are charged to consumers as they leave the store, based on item detection as they pick items up. Studies by C.N. Yogalakshmi and Vivek Maik (2020) discuss the reduction in wait time and overall consumer satisfaction due to the removal of the traditional checkout line. However, there are barriers to the use of RFID and weight sensors in retail settings. The early adopters face an increased, significant upfront price, along with a few accuracy challenges associated with the sensor as well as consumer worry about privacy; RFID tags could reduce friction at checkout along with automated and efficient tracing products, but some consumers do not like the idea or feel comfortable being tracked. Weight sensors have their own accuracy challenges and can be unreliable due to calibration concerns when similarly weighted items are on the same sensor. Even with these barriers, the future scope of RFID and weight sensor-based systems look bright.

The advancement of AI and machine learning could provide notable advances in accuracy and customer safety, as well as improve the total shopping experience. Furthermore, the development of 5G technology could significantly improve the speed of data transmission and eventually aid in developing a real-time scope of these systems. Some researchers such as Peeyush Garg and other (2022) believe the future will inevitably become commonplace with these types of technologies, as consumer appetites only grow for increased speed and frictionless shopping experiences on retail shopping floors.

2.2 EM-18 vs RC522 RFID modules

EM-18:

1) Operates at a lower frequency of typically 125 kHz.

2) Consumes less power, making it suitable for mobile applications.

3) Simple to interface with microcontrollers due to fewer complexities in its communication protocols.

4) Best suited for simple tasks such as identification tasks (scanning tags for inventory management and other low-speed applications).

RC522:

1) Operates at a higher frequency of typically 13.56 Mhz.

2) Consumes slightly more power, especially at higher frequencies.

3) Requires more complex wiring and communication protocols (SPI) to interface with microcontrollers.

4) Better for complex applications requiring higher data throughput or security.

2.2.1 Principles of operation

The EM-18 RFID reader operates on the principle of electromagnetic induction and is used to identify the products that are scanned in the cart. The EM-18 continuously emits a low frequency electromagnetic field through its antenna and operates at a low frequency of 125 kHz. Each product in the store contains a RFID tag that contains a unique identification number. When a tagged item is brought within the vicinity of the EM-18 reader, the tag is powered by the electromagnetic field and transmits its ID back to the reader. This unique ID is then sent via UART communication to the ESP32 which is interfaced with the EM-18 module. Once the microcontroller receives the tag ID, it compares it with entries that are stored in a database that contains all product details such as item names, prices, and standard weights. The system then uses this data to cross-reference the item with a connected weight sensor to ensure accuracy and prevent item mismatch or theft. The product's information is displayed on an LCD screen mounted on the cart, and the item is automatically added to the customer's virtual bill . When the customer finishes shopping, the final bill can be accessed through a local server using the host IP as a login ID, allowing them to complete the transaction directly from the cart

without waiting in traditional billing queues. This process simplifies shopping, improves accuracy, and significantly enhances the overall customer experience.

2.3 Summary and Gaps Identified

Summary

Table 2.1: Summary

Sl No	Item	Description
1	Project name	Smart Billing Cart
2	Microcontroller	ESP-32
3	RFID-reader	EM-18
4	Load Cell	HX711
5	Display	16x2 LCD
2	Database	PHP-based
3	Mobile App	Custom android app
4	Checkout Mechanism	QR code payment
3	Benefits	Faster checkout,user-friendly, increased billing accuracy

Gaps Identified

1) Limited Security for RFID Tags: EM-18 RFID reader can be easily spoofed or cloned, which may compromise item verification and billing integrity.

2) Lack of Real-time Synchronization: If internet or server connectivity fails, the system may not synchronize properly with the PHP-based database.

3) Weight Tolerance and Calibration Issues: HX711 load cell may have minor inaccuracies if not calibrated regularly or if the cart is uneven, leading to verification mismatches.

4) Single Point of Failure (ESP32): ESP32 is responsible for handling all peripherals. If it fails, the entire system halts with no fallback mechanism.

5) Battery and Power Management Not Addressed: Continuous operation of ESP32, LCD, and sensors requires reliable power. Battery drainage or lack of backup isn't discussed.

6) Limited UI on 16x2 LCD: The small display restricts how much information can be shown at a time, possibly affecting user experience.

.

Chapter 3

Methodology

3.1 Problem Statement

In typical retail scenarios, customers sit for a long time waiting to check out which is frustrating. This time delay is the result of cashiers manually scanning items and verifying their prices and weights. This creates a negative experience for customers and is also prone to human error which can potentially increase theft or wrong costs.For the above reasons, we have a need for a smart billing cart system that integrates RFID scanning and verification so that the billing inefficiencies of the traditional method and the theft security risks are addressed.

A smart billing cart equipped with RFID tags and an EM-18 RFID reader would enable customers to scan an item on their own while shopping. The cart would provide real-time product information (price and weight) to an LCD display. The items would be cross-referenced against a database that includes the price and weight of the product linked to the RFID tag. However, to account for security concerns, we would implement an anti-theft feature based upon weight. As the user scans the item, the weight in the database would be cross-referenced against a real-time weight measurement of the item using the cart's weight sensor. If a weight discrepancy occurs – like scanning a less expensive item but weighing a more expensive item – the system would identify the mismatched weight and subsequently, stop the user from using the checkout feature in the app. This theft feature prevents the user from simply swapping out items to shorten the system flow and to ensure that the transaction is consistent, maintaining security purposes within retail transactions.

3.2 Block Diagrams

The block diagram given below illustrates the core components of the Smart Billing Cart system. The ESP32 microcontroller acts as the central unit, receiving data from the EM-18 RFID reader for product identification and from the HX711 load cell for weight measurement. It processes this information and displays relevant output on the 16x2 LCD screen for user feedback.

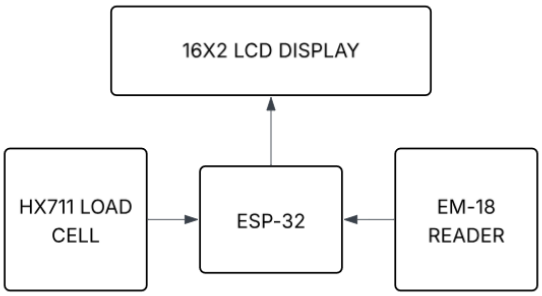


Figure 3.1: Hardware block diagram

The below block diagram shows how both the hardware (ESP32-based system) and software (Android app) communicate with a local server that hosts the database. The hardware and software components send requests to fetch or update product information, and the server responds accordingly. This two-way communication ensures synchronization between physical product handling and digital transaction processing.

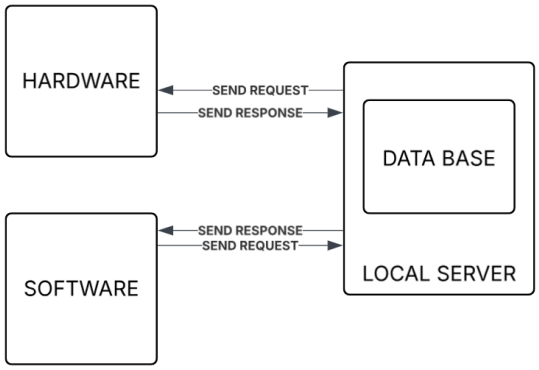


Figure 3.2: System block diagram

3.2.1 Data Flow

The flowchart below outlines the step-by-step data flow of the Smart Billing Cart system:

1)Start RFID Scanning: The process begins when a user scans an RFID tag using the EM-18 reader. The ESP32 receives this data and checks whether the tag has been scanned an odd or even number of times.

2)Item Update in Cart: If scanned an odd number of times, the item is considered added to the cart.If scanned an even number of times, it's treated as removed from the cart. This logic ensures simple toggling without needing separate add/remove actions.

3)Weight Measurement & Validation: After updating the cart, the load cell measures the total weight. This weight is then compared with the expected weight stored in the database for validation.

4)Validation Outcome: If the weight does not match, the checkout is disabled in the app to prevent manipulation.If the weight matches, the product is added to the bill, and checkout is enabled.

5)Payment Process: Finally, a QR code scanner is open for the user to scan and complete the payment, which concludes the transaction.

This system tightly integrates scanning, weight checking, validation, and payment, all flowing through the ESP32 and ensuring secure and efficient billing.

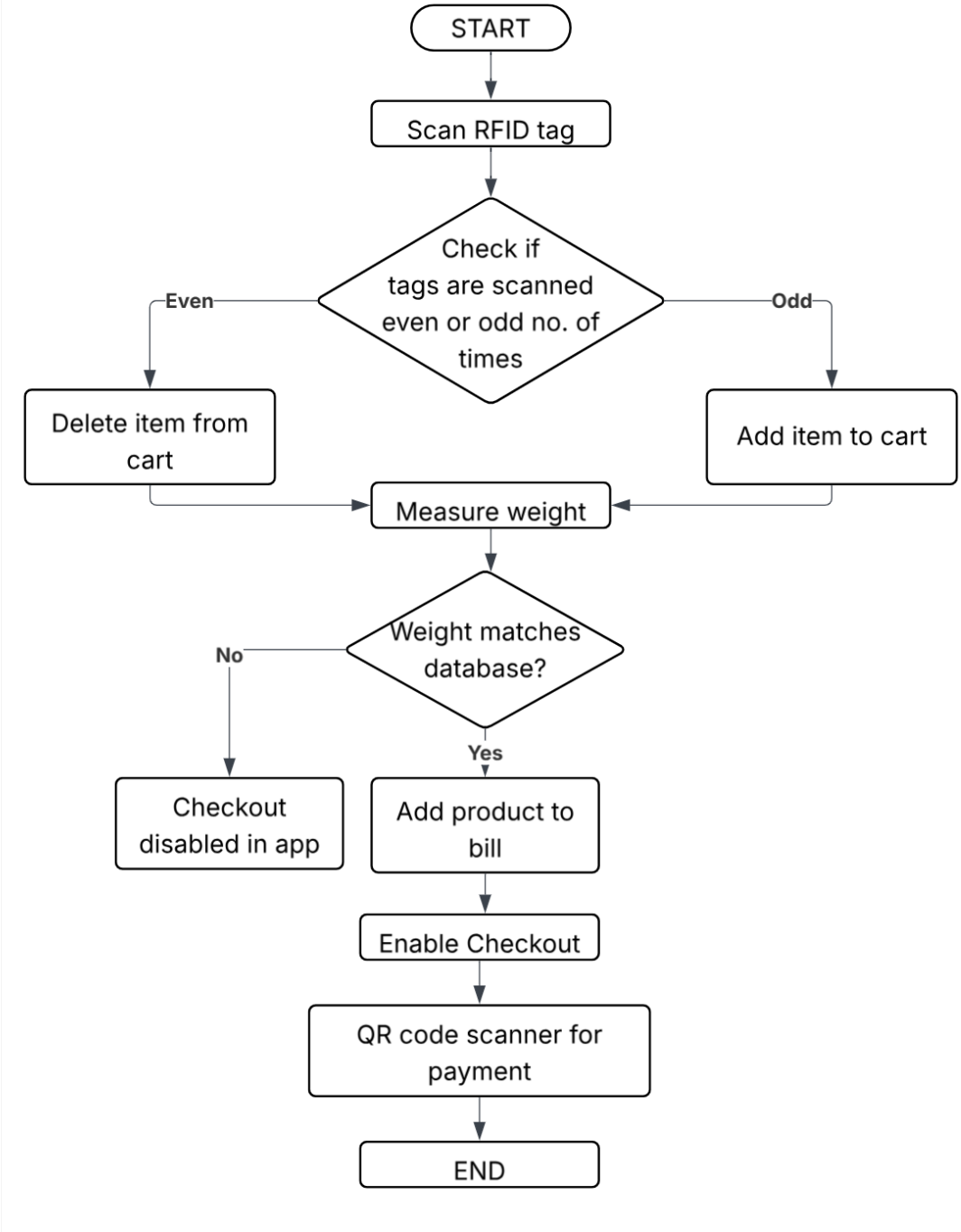


Figure 3.3: Flowchart

3.3 Circuit Diagram

The circuit diagram below shows the hardware setup for a smart cart system using an ESP32 microcontroller. The HX711 load cell amplifier is connected to the ESP32 through GPIO pins 27 (Data) and 26 (SCK), allowing for weight measurement via a strain gauge-based load cell. The EM-18 RFID reader communicates with the ESP32 using UART protocol, with the reader's Tx connected to GPIO pin 34 (ESP32's Rx) to receive RFID tag data. A 16x2 I2C LCD is interfaced using the I2C protocol, connected via GPIO pins 36 (SCL) and 33 (SDA), to display item and weight details. Power is supplied using a 5V line, with common GND connections to all modules ensuring a stable system. This setup enables RFID-based item identification, weight validation, and real-time data display on the LCD. ESP-32 is given a power of 3.3V with the help of resistors.

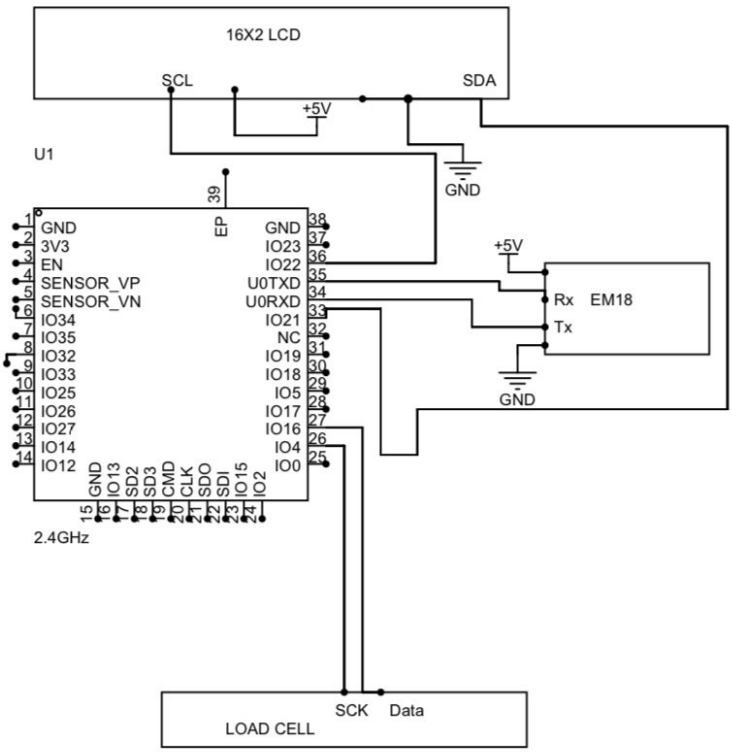


Figure 3.4: Circuit Diagram

The block diagrams and flowcharts collectively illustrate the seamless integration of hardware and software components in the smart cart system.

3.4 Estimate of product/ Product cost

Table 3.1: Components and its costs

Sl No	Component	Cost
1	ESP-32 WROOM 32	Rs.700
2	16X2 LCD DISPLAY	Rs.120
3	I2C LCD INTERFACE MODULE	Rs.150
4	STRAIN GAUGE-BASED LOAD CELL	Rs.125
5	HX711 MODULE	Rs.200
6	EM-18 READER MODULE	Rs.275
7	EM18 RFID READER BOARD	Rs.200
8	RFID TAGS	Rs.20/piece
9	EXTRAS	Rs.630

TOTAL COST : Rs.2500

Chapter 4

Embodiment and Detailed Design

This chapter breaks down how the Smart Billing Cart was actually brought to life—right from picking out the components to getting them all working together. It walks through both the hardware setup and the software part, showing how everything connects, from the cart to the mobile app and the database. The idea is to clearly show how each piece plays its role and how they all work in sync to make the whole system run smoothly.

4.1 Working Principle:

When a product is placed in the cart, its RFID tag is scanned by the EM-18 reader and its details are fetched from the database. The item is then weighed using the HX711 load cell, and the measured weight is compared with the stored value to verify its authenticity. If the weights match, the item is added to the bill and the checkout option in the app is enabled; otherwise, checkout remains disabled to prevent mismatches or errors.

4.2 System Level Design

4.2.1 Components Used

1) ESP-32 WROOM 32
2) 16X2 LCD DISPLAY
3) I2C LCD INTERFACE MODULE
4) STRAIN GAUGE-BASED LOAD CELL
5) HX711 MODULE
6) EM-18 READER MODULE
7) EM18 RFID READER BOARD
8) RFID TAGS

4.2.2 Functions of each component

ESP-32 WROOM 32

The ESP32 is a compact and efficient microcontroller with built-in Wi-Fi and Bluetooth, and it basically acts as the brain of the Smart Billing Cart. It takes input from both the RFID reader and the weight sensor, processes these data, and communicates with the mobile app and the database.

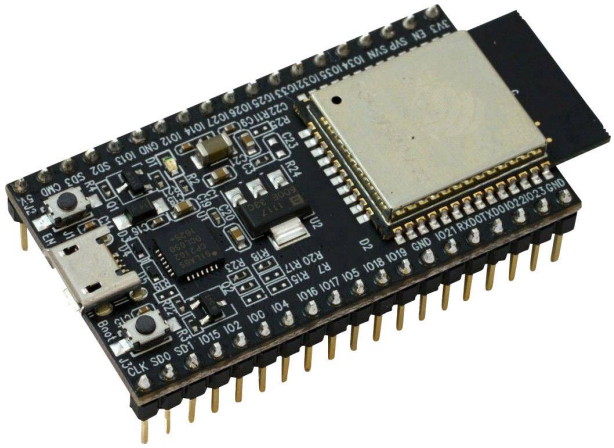


Figure 4.1: ESP32-Wroom 32

16X2 LCD DISPLAY

A 16x2 LCD display is a basic alphanumeric display module capable of showing 16 characters on each of its two lines. It operates by receiving data and commands from a microcontroller, which controls what is displayed. The LCD is commonly used in embedded systems to show real-time information such as messages, status updates, or sensor values.

Figure 4.2: 16X2 LCD Display

I2C LCD INTERFACE MODULE

An I2C LCD interface module is an add-on board that allows an LCD display to communicate using the I2C protocol. It reduces the number of pins needed to connect the LCD to a microcontroller from 16 to just 2 (SDA and SCL). This makes wiring simpler and frees up pins for other components in embedded systems.

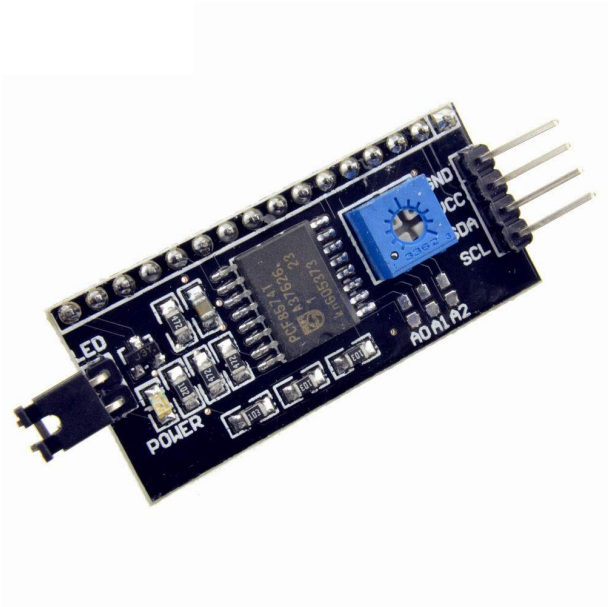


Figure 4.3: I2C LCD interface module

STRAIN GAUGE-BASED LOAD CELL

A strain gauge-based load cell is a sensor used to measure weight by detecting how much a metal part bends under pressure. When an item is placed on it, the metal flexes just a little, and the strain gauge stuck to it changes its electrical resistance, changing into a voltage signal and then processed to figure out the exact weight.

Figure 4.4: Strain gauge-based load cell

HX711 MODULE

The HX711 module is a small chip that helps read weight from a load cell more accurately. It takes the tiny electrical signal produced by the load cell, boosts it, and turns

it into a digital value that a microcontroller (like ESP32). It's especially useful because it can detect even very small changes in weight, making it perfect for electronic weighing systems..

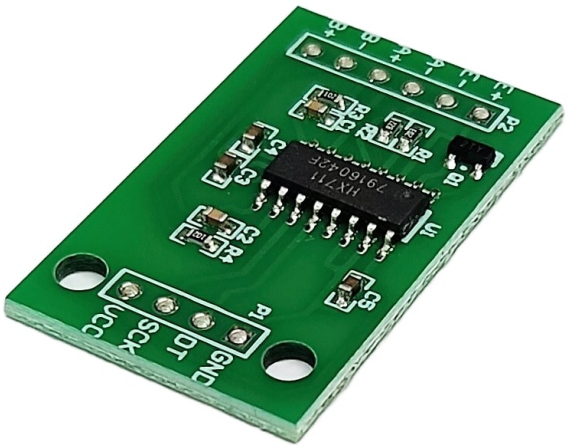


Figure 4.5: HX711 Module

EM-18 READER MODULE

The EM-18 reader module is an RFID reader used to detect and read data from RFID tags. When a tag is brought near the reader, it picks up the unique ID stored in the tag and sends it to a microcontroller like the ESP32. It's commonly used in access control, attendance systems, and smart carts because it's fast, reliable, and easy to use.

Figure 4.6: EM-18 Reader module

EM-18 RFID READER BOARD

The EM18 RFID Reader Board is a simple and popular device used to read RFID tags, which are often found in things like ID cards, key fobs, or smart labels. It is working by emitting a low-frequency radio signal (125 kHz) that powers up the RFID tags nearby. When the tag is close enough, it sends back its unique ID number, which the reader picks up and sends to a connected system like a micro controller. The range of EM18 is around 5 to 10 cm, so the reader can work best for short-range applications. It is easy to integrate these into projects for things like access control, asset tracking, or even library systems where books are tagged and checked in or out. It is a simple way to scan and interact with RFID tags in a variety of real-world applications.

Figure 4.7: EM-18 RFID Reader board

RFID TAGS

RFID tags used with the EM-18 reader are small, wireless devices that store a unique identification number. When brought near the reader, radio waves at a frequency of 125 kHz, allowing the system to identify the item they're attached to. These tags are lightweight, affordable, and commonly used in smart carts, access control systems, and inventory management because they're simple and efficient.

Figure 4.8: EM-18 Tags

4.2.3 Softwares Used

1) **ARDUINO IDE**

The Arduino Integrated Development Environment - or Arduino Software (IDE) - 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. Programs written using Arduino Software (IDE) are called Sketches. These sketches are written in the text editor and are saved with the extension '.ino'. The editor 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 Software (IDE), including complete error messages and other information. The bottom right-hand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open and save sketches and open the serial monitor.

Figure 4.9: Arduino Logo

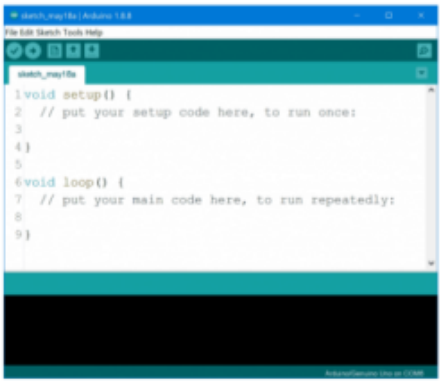


Figure 4.10: Arduino Software Window

2)**XAMP**

XAMPP is a free and easy tool that lets you set up a local server on your computer for web development. It bundles Apache (a web server), MySQL (a database), PHP (for dynamic websites), and Perl (another programming language) all in one package. The control panel is user-friendly, so you can start and stop things like Apache and MySQL with a single click. You get phpMyAdmin to manage your databases easily. Whether you're on Windows, macOS, or Linux, XAMPP works on all of them, making it simple to test your website locally before it goes live. It's an easy way for developers to build and test their projects without needing an external host.XAMPP and IP addresses are related because XAMPP uses your computer's IP address to make your local server accessible. When you run XAMPP, Apache (the web server) listens for requests from your browser, and you can access your local site using the IP address or localhost. These both refer to your own computer allowing you to test and develop websites locally.

If you want others on your local network to access your website, you can use your computer's local IP address. By entering this IP address in a browser, people on the same

network can view your website. However, this won't work over the internet unless you configure your router for external access, which requires port forwarding and setting up a public IP address. In short, XAMPP uses IP addresses to serve your site locally (via localhost or your local network IP address') and is the backbone for testing sites before making them publicly accessible.

Figure 4.11: Xamp Logo

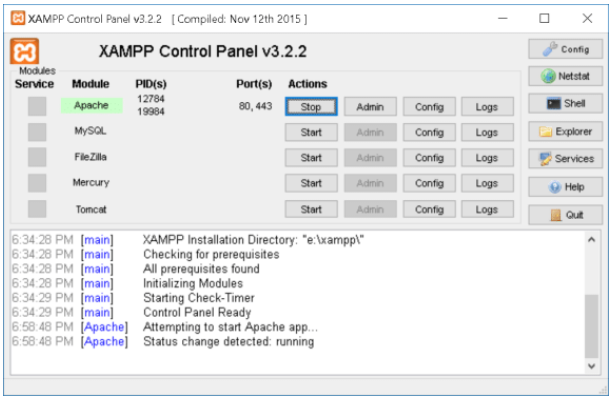


Figure 4.12: Xamp control panel

3)**ANDROID STUDIO**

Android Studio is the main tool used for building Android apps. It's like a workspace where you can write, test, and debug your code all in one place. The code editor in Android Studio helps you by highlighting important parts of your code and even suggests fixes or improvements while you're typing. There's a message area that gives you feedback as you build and run your app, showing whether things went well or if there were any errors. The console below shows any log messages or debugging info, which is super helpful when you're troubleshooting your app.

The toolbar at the top has buttons for running and testing your app, so you can see how it looks and behaves on a virtual device or an actual phone. You can manage all your

project files, like your code and design layouts, in the file explorer, and they're saved with specific extensions like .java for code or .xml for layout files. Android Studio also makes it easy to connect your app to tools like Firebase or Google Play, and it even lets you simulate how your app will work on different devices using an emulator. The layout editor allows you to visually design your app's interface, dragging and dropping elements instead of just coding everything by hand. It's a comprehensive tool that makes the process of developing Android apps easier and more streamlined.

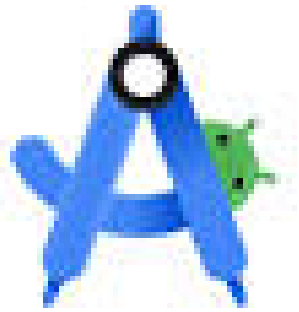


Figure 4.13: Android Studio Logo

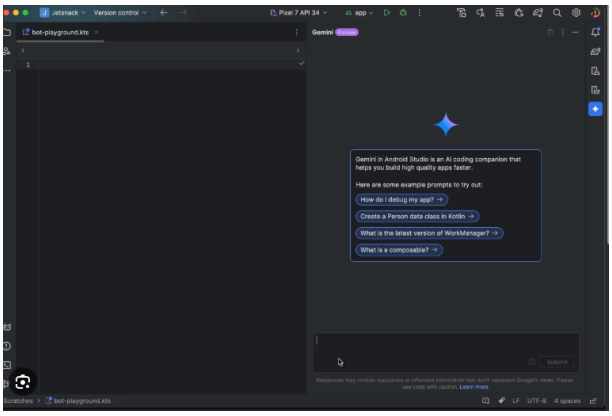


Figure 4.14: Android Studio Interface

Chapter 5

Results and Discussions

In this project, product scanning and billing were successfully carried out using an ESP32 microcontroller, along with an EM-18 RFID reader and HX711 load cell. The system was able to identify products accurately by reading RFID tags, then verifying their weight with the sensor and comparing it with values stored in the php database. Only when the weight matched with the stored data, the checkout button in the Android app was enabled, ensuring a secure billing process.

The communication between the ESP32 and the Android application was smooth and responsive, allowing real-time updates and user interaction. The 16x2 LCD display provided immediate feedback for each action such as item scanned, matched, or removed thus improving the overall user experience. The integration of hardware and software worked well, and the system proved reliable during testing with multiple items.

This experiment shows that smart billing carts can significantly reduce checkout times while increasing accuracy and security. Further enhancements, such as adding image recognition or better weight calibration, could make the system even more efficient and scalable for retail environments.

5.1 Final working model status

The final working model of the smart cart system operates smoothly, integrating RFID tag scanning, weight measurement, and LCD display functions effectively. All hardware components communicate reliably with the ESP32, and the system accurately adds or removes items while validating weights against the database. The checkout and billing process is successfully managed through QR code scanning, ensuring a streamlined user experience.

5.2 Photos of the product demo

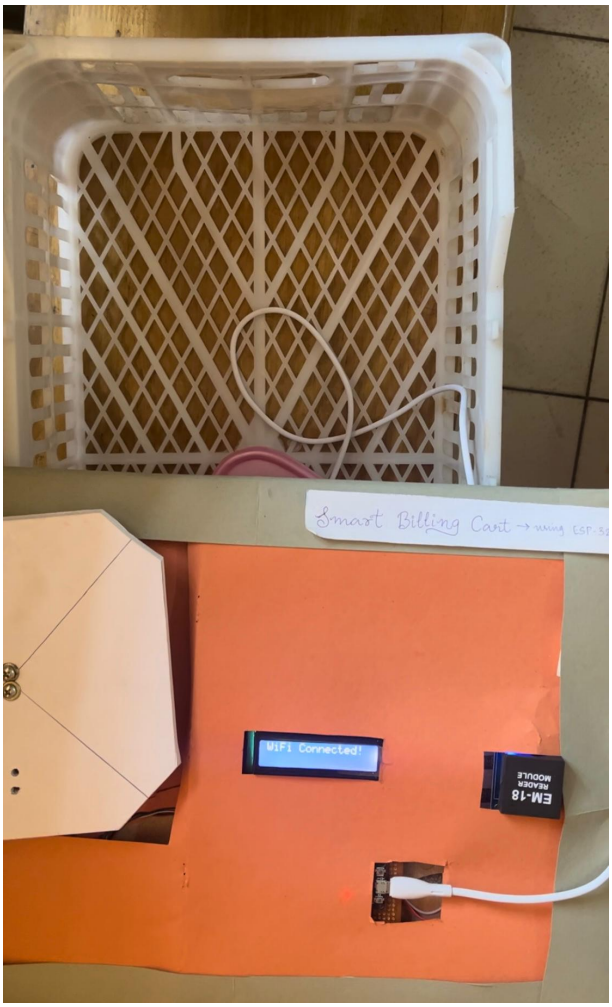


Figure 5.1: Smart billing cart connected to WiFi

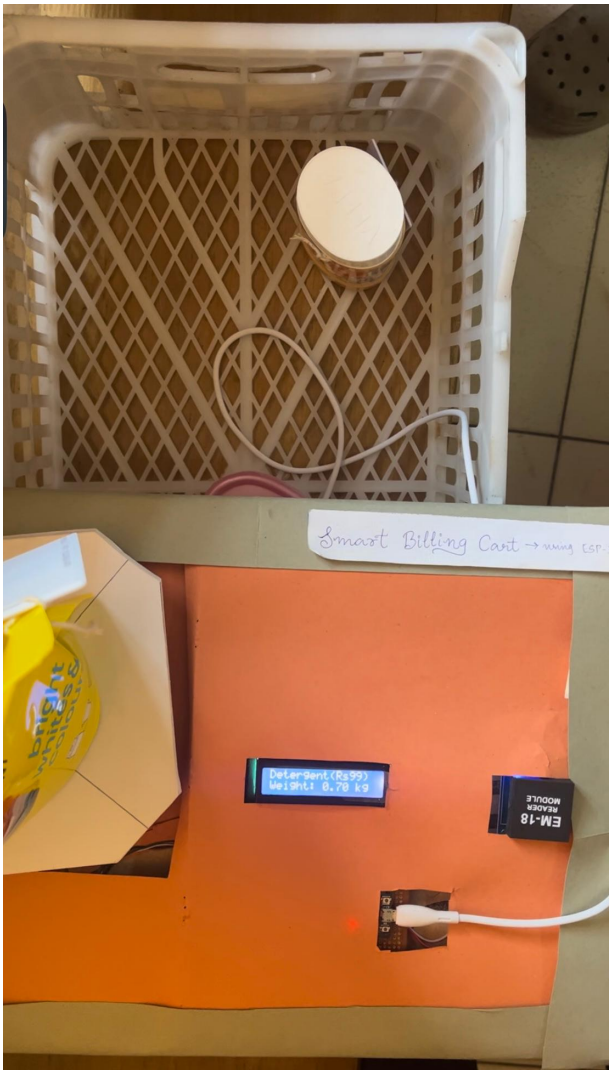


Figure 5.2: Items being scanned and added to cart

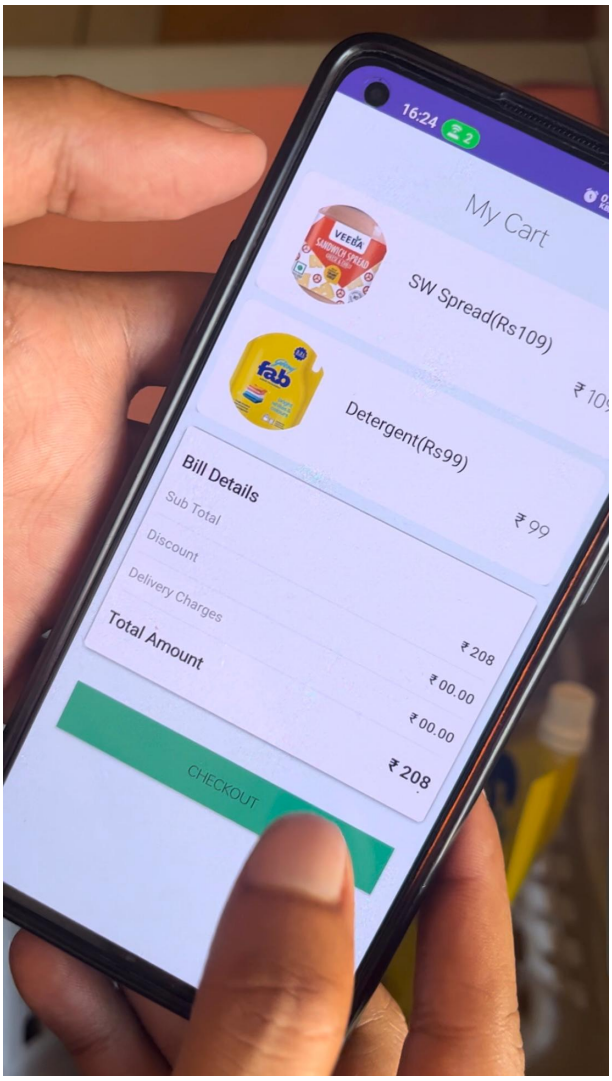


Figure 5.3: Checkout button enabled since no mismatch of weight

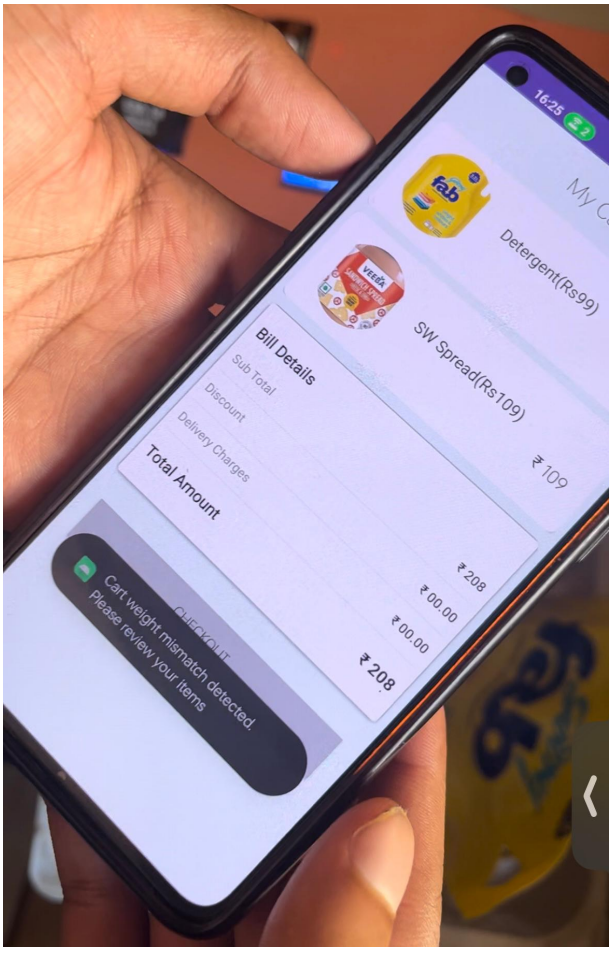


Figure 5.4: Checkout button disabled since mismatch of weight occured

5.2.1 Challenges faced

1) **RFID Tag Detection Errors** - Sometimes the EM-18 RFID reader may fail to detect tags quickly or accurately, especially if tags are stacked or shielded by other materials.
2) **Load Cell Inaccuracy** - The HX711 module needs precise calibration. Any imbalance or surface vibration can affect weight readings.
3) **Data Syncing with PHP Server** - Ensuring real-time and reliable communication between the ESP32 and the PHP-based server/database can be tricky, especially on unstable Wi-Fi.
4) **Power Management** - Maintaining stable power supply to ESP32, RFID, LCD, and Load Cell components was a challenge, especially if running on battery.
5) **Component Integration** - Making all components (RFID, load cell, LCD) work seamlessly with ESP32 required careful wiring and coding.
6) **Android App Communication** - Ensuring smooth Bluetooth/Wi-Fi communication between the Android app and the ESP32 with no lag or disconnection issues.
7) **Product Weight Variation** - Slight changes in packaging or product weight led to false mismatch errors during weight verifications.

5.2.2 Work division

Table 5.1: Work Division

Sl No	NAME	WORKDONE
1	KEZIAH MARIAM VARUGHESE	Software and Data base creation
2	K. VIJAY KRISHNAN	Hardware connections,design of app
3	MARIA SIJU JOSEPH	Software,Adding products to Data base
4	MILAN THOMAS C	Hardware connections, design of app

Chapter 6

Conclusions & Future Scope

The Smart Billing Cart project successfully demonstrates an innovative approach to automating the retail checkout process. By integrating the ESP32 microcontroller with the EM-18 RFID reader, HX711 load cell, and a 16x2 LCD, the system ensures real-time product identification and weight verification. The use of a PHP-based backend and a custom Android application streamlines billing and enables secure QR code payments only after all items are authenticated.

This smart system not only enhances the speed and accuracy of shopping but also adds an extra layer of security and convenience for customers. Overall, the project provides a scalable solution to reduce manual errors, eliminate long queues, and improve the overall shopping experience. It sets the foundation for smart retail environments and opens the door for further enhancements in automation and customer service.

In the future, there's a lot we can build on to make the Smart Billing Cart even better. Imagine if the cart could not only scan products with RFID but also recognize them visually using a camera and AI—that would help catch any errors or mismatches more easily. Instead of relying on a local server, we could move everything to the cloud, which would make the system faster and easier to manage across multiple stores. The mobile app could also be improved by adding features like special offers for users, points for frequent shoppers, or even multiple language options to make it more inclusive. We could also add smart sensors or cameras to detect if someone tries to remove an item without scanning, adding an extra layer of security. And to top it off, adding more flexible payment options—like using different apps, digital wallets, or even voice commands—would make the checkout process super smooth and convenient for everyone.

References

[1] A. Laxmi, B. Shraddha, A. Chothave, and P. Samreen, "Smart shopping cart using rfid technology," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 7, no. 11, pp. 146–150, Nov. 2018.

[2] G. M. Rao, K. Preethi, A. S. Krishna, A. Firdaus, and C. Lokesh, "Rfid based smart trolley for automatic billing system," *International Journal of Recent Technology and Engineering (IJRTE)*, vol. 9, no. 1, pp. 1593–1597, May 2020.

[3] B. S. Vyshnavi, S. N. Kishor, and G. N. K. Ramaiah, "Electronic shopping cart," *International Journal of Engineering Research & Technology (IJERT)*, vol. 9, no. 12, pp. 1–5, Dec. 2020.

[4] M. T. Dangat, P. Pawar, S. Vaidya, P. Khaire, and A. Randive, "Smart trolley with automatic billing system using rfid," *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, 2025.

[5] C. N. Yogalakshmi and V. Maik, "Innovative automated shopping trolley with rfid and iot technologies," in *Artificial Intelligence and Evolutionary Computations in Engineering Systems*. Singapore: Springer, 2020, available on SpringerLink.

[6] P. Garg, T. Joshi, and D. Bansal, "Design and development of rfid based smart shopping cart using arduino," *International Journal of Electronic Commerce Studies*, 2022, available via Journals at ATISR.

Appendix A: Program

```
   #include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <WiFi.h>
#include <HTTPClient.h>
#include <HX711.h>

// Wi-Fi Credentials
const char* ssid = "OnePlus 9r";
const char* password = "aaaaaaaa";
String IPV4 ="192.168.165.60";
String serverURL = "http://" + IPV4 + "/blue_shopping_cart/blue_shopping_cart_3.php"

// LCD Setup (I2C Address: 0x27)
LiquidCrystal_I2C lcd(0x27, 16, 2);

// RFID Scanner
HardwareSerial RFID(1);  // RFID Module
#define RFID_RX 19
#define RFID_TX 18

// Load Cell (HX711)
#define LOADCELL_DOUT 16
#define LOADCELL_SCK 4
HX711 scale;

// RFID Product Codes
String rfid_P1 = "270018450E74";
String rfid_P2 = "27001843502C";
String rfid_P3 = "270018441E65";
String rfid_P4 = "27001845DEA4";

// Product Names
```

```
String productNames[] = { "SW Spread(Rs109)", "Gift Cup(Rs700)", "Org SMint(Rs325)",
String rfidCodes[] = { rfid_P1, rfid_P2, rfid_P3, rfid_P4 };

float calibration_factor = 443000; // Adjust based on calibration
String lastRFID = "";
unsigned long lastRFIDTime = 0;
const int scanInterval = 3000;  // Prevent multiple scans within 3 seconds

void setup() {
    Serial.begin(115200);

    // Initialize RFID
    RFID.begin(9600, SERIAL_8N1, RFID_RX, RFID_TX);  // RFID Scanner

    // Initialize LCD
    lcd.begin();
    lcd.backlight();
    lcd.setCursor(0, 0);
    lcd.print("Smart Cart Ready");
    delay(2000);

    // Initialize Wi-Fi
    connectWiFi();

    // Initialize Load Cell
    scale.begin(LOADCELL_DOUT, LOADCELL_SCK);
    scale.set_scale(calibration_factor);   // Set calibration factor
    scale.tare();                          // Automatically tare the scale (reset to

    Serial.println("ESP32 Smart Shopping Cart Ready...");
}
```

```
void connectWiFi() {
    WiFi.begin(ssid, password);
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Connecting WiFi...");
    int attempts = 0;

    while (WiFi.status() != WL_CONNECTED && attempts < 20) {
        delay(500);
        Serial.print(".");
        attempts++;
    }

    if (WiFi.status() == WL_CONNECTED) {
        Serial.println("\nWiFi Connected!");
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("WiFi Connected!");
        delay(2000);
    } else {
        Serial.println("\nWiFi Failed!");
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("WiFi Failed!");
    }
}

// Function to Send Data to Server
void sendToServer(String rfid, String product, int qty, float weight, String lastIte
    if (WiFi.status() != WL_CONNECTED) {
        Serial.println("WiFi disconnected, attempting to reconnect...");
        connectWiFi();
```

```
    }

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

        // Send 'lastItem' (the last scanned product)
        String postData = "rfid=" + rfid + "&product=" + product + "&quantity=" + St
        int httpResponseCode = http.POST(postData);

        if (httpResponseCode > 0) {
            // Get the response body as a string
            String response = http.getString();
            Serial.println("Sent to server: " + postData + "\n" + response );
        } else {
            Serial.println("Failed to send data. HTTP error: " + String(httpResponse
        }

        http.end();
    } else {
        Serial.println("WiFi not connected!");
        lcd.setCursor(0, 1);
        lcd.print("WiFi Disconnected");
    }
}

void readRFID() {

    if (millis() - lastRFIDTime < scanInterval) {
        return; // Debounce RFID scans
    }
```

```
String rfid_data = RFID.readStringUntil('\n');
rfid_data.trim(); // Remove unwanted characters

if (rfid_data.length() > 0) {
    Serial.println("RFID Scanned: " + rfid_data);
    lastRFIDTime = millis();
    lastRFID = rfid_data;

    String lastItem = getProduct(rfid_data);
    if (lastItem == "") {
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Unknown Item");
        Serial.println("Unknown RFID scanned.");
        return;
    }

    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Place Item...");

    delay(2000);  // Wait before measuring weight

    float weight = measureWeight();

    // Pass 'lastItem' to the sendToServer function
    sendToServer(rfid_data, lastItem, 1, weight, lastItem);

    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print(lastItem);
```

```
        lcd.setCursor(0, 1);
        lcd.print("Weight: " + String(weight) + " kg");
        delay(3000);

        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Scan Product...");
    }
}

String getProduct(String rfid) {
    for (int i = 0; i < 4; i++) {
        if (rfid == rfidCodes[i]) {
            return productNames[i];
        }
    }
    return "";
}

float measureWeight() {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Measuring...");

    delay(1000); // Stabilize the sensor before reading

    float weight = scale.get_units(10);
    Serial.println("Measured Weight: " + String(weight) + " kg");
    return weight;
}

void loop() {
```

```
    readRFID();
}
```

Appendix B: Presentation Slides

FINAL PRESENTATION

AUTOMATIC BILLING CART

GUIDE : MS. SONI MONICA

TEAM MEMBERS:
KEZIAH MARIAM VARUGHESE
K VIJAY KRISHNAN
MARIA SIJU JOSEPH
MILAN THOMAS C

▶ MINI PROJECT-ABSTRACT

INTRODUCTION

Automatic Billing Cart Concept:

- Integrates RC-522 Reader and ESP32 to automate product selection, billing, and checkout.
- Eliminates manual scanning at checkout, providing a smooth retail experience.

ESP32 as the Central Processor:

- The ESP32 microcontroller processes data from the RC-522 reader, product database, and user interface.
- It ensures efficient communication and control over the system's operations, including managing Wi-Fi connections.

INTRODUCTION

RC-522 for Product Identification:

- RFID tags on products are detected by an RFID reader as items are added to the cart.
- The ESP32 processes the data to identify products and retrieve relevant details (e.g., price, description,weight) from a database.

Seamless Checkout Process:

- The system automatically calculates the total bill based on scanned items.
- Payment can be processed via mobile apps or integrated payment systems, streamlining the checkout process.

▶ MINI PROJECT-ABSTRACT

OBJECTIVES

1)Enhance Shopping Experience

2)Enable Contactless Payment

3)Promote Sustainability

COMPONENTS REQUIRED

Hardware Specifications

- ESP 32 Microcontroller
- RC522 Module & Cards
- LCD Display
- PCB and Breadboards
- Cables and Connectors
- Adapter
- Weight sensor (HX711)

▶ MINI PROJECT-ABSTRACT

COMPONENTS REQUIRED

Software Specifications

ARDUINO-IDE

XAMP

ANDROID STUDIO

METHODOLOGY

RFID Product Identification:

- Each product has an RFID tag for identification.

Scanning Process:

- The customer swipes the product RFID card; the reader sends the ID to the ESP32.

Price Retrieval:

- The microcontroller fetches the product price from a database or memory.

▶ MINI PROJECT-ABSTRACT

METHODOLOGY

Total Calculation:

- The total cost is updated as each product is scanned.

Bill Finalization:

- The user presses a "Finish" button to finalize the bill.

Display and Payment:

- The total amount is shown on an LCD, and the payment process is triggered.

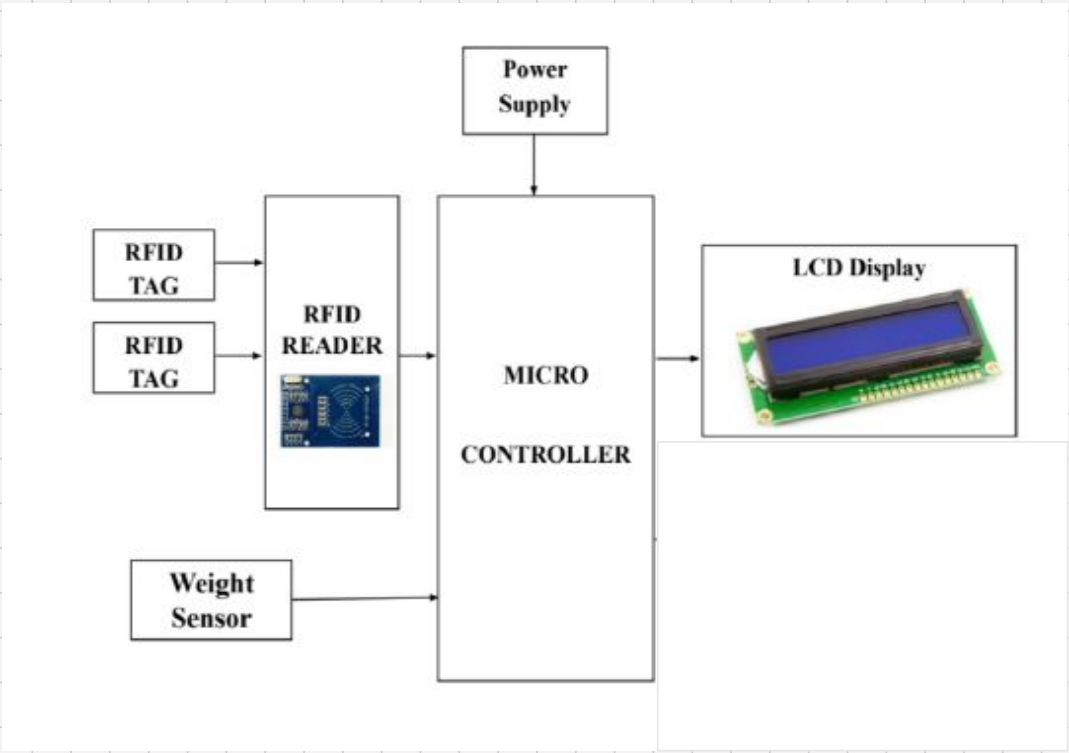
ANTI-THEFT

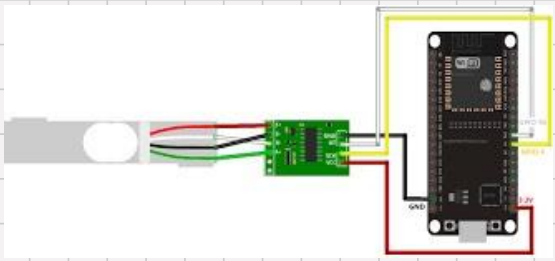
Objective: The anti-theft system is designed to ensure that customers only pay for the items they've actually picked up in an automatic billing cart.

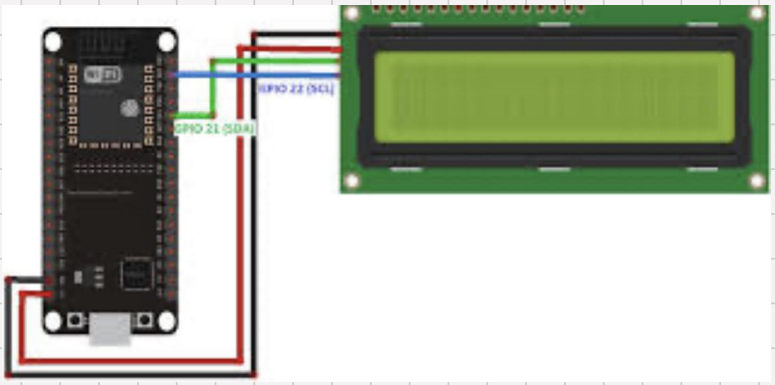
RFID Integration: Each product in the cart is associated with an RFID tag that stores the weight of the item. When a customer places an item in the cart, the RFID scanner reads the tag and stores the item's weight data in the system.

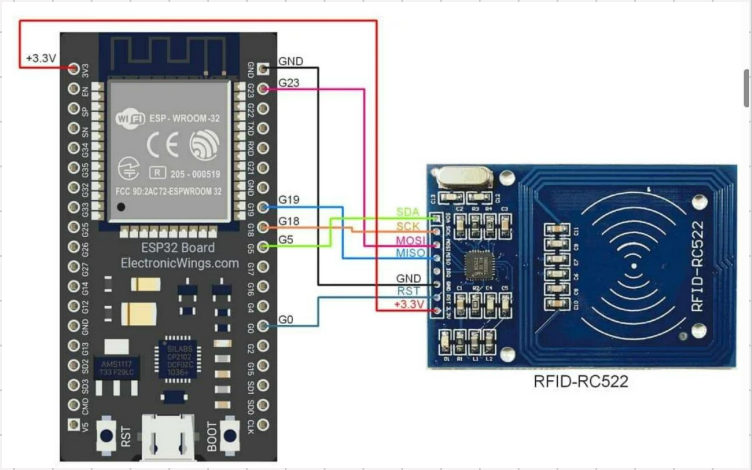
Weight Sensor: The cart is equipped with a weight sensor (HX711), which continuously measures the total weight of the items inside the cart. The HX711 is an analog-to-digital converter that provides precise weight measurements by interfacing with load cells.

Theft Detection: If there's a mismatch in weight (i.e., the RFID tag's recorded weight is less than the actual weight measured by the sensor), the system triggers an alert.









MINI PROJECT-ABSTRACT

REFERENCES

1) SB, Mr. (2024). SMART SHOPPING CART USING ESP. INTERANTIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT. 08. 1-5. 10.55041/IJSREM32440.

2) Prerana, Thakur, Shikha Ranjan, and Prachi Kaushik. "Smart Shopping Cart for Automatic Billing in Supermarket." *International Journal of Engineering Development and Research* 5.2 (2017): 975-978.

3) Dhande, Pratiksha, Chanchal Gupta, and Pushpendra Singh Chouhan. "RFID BASED AUTO BILLING SHOPPING CART."

Appendix C: Poster

RSET
RAJAGIRI SCHOOL OF ENGINEERING & TECHNOLOGY

SMART BILLING CART

GUIDE : MS. SONI MONICA

TEAM MEMBERS -S6 ECE(B)

KEZIAH MARIAM VARUGHESE (U2201117)
K VIJAY KRISHNAN (U2201115)
MARIA SIJU JOSEPH (U2201126)
MILAN THOMAS C (U2201133)

OBJECTIVES

1) Automated Product Identification and Weight Verification
2)Real-Time Cart Monitoring and Billing
3)Anti-Theft Detection System

INTRODUCTION

The rapid growth of smart retail technologies has led to the development of intelligent systems that enhance the shopping experience and improve security. A smart trolley equipped with RFID and weight-sensing technology offers automated billing, real-time monitoring, and theft prevention. This project utilizes an ESP32 microcontroller, EM-18 RFID reader, and HX711 load cell to create a cost-effective, efficient, and secure shopping solution

METHODOLOGY

The smart trolley system integrates an ESP32 microcontroller with an EM-18 RFID reader to scan product tags and a HX711 load cell to measure the weight of each item. Each scanned product's weight is verified against a backend database to ensure accuracy and authenticity.

FINISHED PRODUCT

RESULTS

1. Accurate Product Verification: The system successfully identified items using RFID and verified their weights against the database with high accuracy.
2. Efficient Billing System: Real-time tracking and automatic billing of products were achieved, reducing manual errors and checkout time.
3. Effective Anti-Theft Mechanism: The trolley detected discrepancies between scanned items and actual weight, triggering alerts to prevent theft or unbilled items.

CONCLUSION

The smart trolley system effectively streamlines the shopping process by automating billing, ensuring product authenticity, and enhancing security through real-time weight verification and anti-theft detection.

Appendix D: Vision,Mission,Programme Outcomes and Course Outcomes

Vision, Mission, Programme Outcomes and Course Outcomes

Institute Vision

To evolve into a premier technological institution, moulding eminent professionals with creative minds, innovative ideas and sound practical skill, and to shape a future where technology works for the enrichment of mankind.

Institute Mission

To impart state-of-the-art knowledge to individuals in various technological disciplines and to inculcate in them a high degree of social consciousness and human values, thereby enabling them to face the challenges of life with courage and conviction.

Department Vision

To evolve into a Center of Excellence in Electronics Communication Engineering, moulding Professionals having Inquisitive, Innovative and Creative minds with sound practical skills who can strive for the betterment of mankind.

Department Mission

To impart state-of-the-art knowledge to students in Electronics Communication En- gineering and to inculcate in them a high degree of social consciousness and a sense of human values, thereby enabling them to face challenges with courage and conviction.

Programme Outcomes (PO)

Engineering Graduates will be able to:

1. Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2.Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. Conduct investigations of complex problems: Use research-based knowledge including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. Modern Tool Usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and Team work: Function effectively as an individual, and as a member or leader in teams, and in multidisciplinary settings.

10. Communication: Communicate effectively with the engineering community and with society at large. Be able to comprehend and write effective reports documentation. Make effective presentations, and give and receive clear instructions.

11. Project management and finance: Demonstrate knowledge and understanding of engineering and management principles and apply these to one's own work, as a member and leader in a team. Manage projects in multidisciplinary environments.

12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change.

Programme Specific Outcomes (PSO)

PSO1: Demonstrate their skills in designing, implementing and testing analogue and digital electronic circuits, including microprocessor systems, for signal processing, communication, networking, VLSI and embedded systems applications.

PSO2: Apply their knowledge and skills to conduct experiments and develop applications using electronic design automation (EDA) tools.

PSO3: Demonstrate a sense of professional ethics, recognize the importance of continued learning, and be able to carry out their professional and entrepreneurial responsibilities in electronics engineering field giving due consideration to environment protection and sustainability

Course Outcomes (CO)

CO 1: Students will be able to practice acquired knowledge within the selected area of technology for project development.

CO 2: Students will be able to Identify, discuss and justify the technical aspects and design aspects of the project with a systematic approach.

CO 3: Students will be able to Reproduce, improve and refine technical aspects for engineering projects.

CO 4: Work as a team in development of technical projects

CO 5: Communicate and report effectively project related activities and findings.

Mini Project Outcomes

Mini Project Outcome 1: Designed and developed an embedded system using ESP32

Mini Project Outcome 2: Justified the selection of components and followed a structured approach

Mini Project Outcome 3: Refined the system through calibration and optimization improvements

Mini Project Outcome 4: Collaborated effectively as a team to integrate hardware and software components

Mini Project Outcome 5: Documented the project developments through reports and presentations.

Appendix E: CO-PO-PSO Mapping

CO - PO Mapping

	PO 1	**PO 2**	**PO 3**	**PO 4**	**PO 5**	**PO 6**	**PO 7**	**PO 8**	**PO 9**	**PO 10**	**PO 11**	**PO 12**
CO 1	3	3	3	2	3	3	2					2
CO 2	3	3	3	2	3	3	2			3	2	2
CO 3	3	3	3	2	3	3	2			3	2	2
CO 4					2			3	3	3	2	2
CO 5					2			3	3	3	2	2

CO - PSO Mapping

	PSO 1	**PSO 2**	**PSO 3**
CO 1	3	3	2
CO 2	3	3	2
CO 3	3	3	2
CO 4	2	2	3
CO 5	2	2	3

Mini Project Outcomes - PO Mapping

	PO 1	**PO 2**	**PO 3**	**PO 4**	**PO 5**	**PO 6**	**PO 7**	**PO 8**	**PO 9**	**PO 10**	**PO 11**	**PO 12**
P1	3	2	2			2	2					2
P2	3		2			1	2			3	2	2
P3	3	3	1				2			3	2	2
P4								2	3	3	2	2
P5								2	3	3	2	2

Justifications for Mini Project Outcome - PO Mapping

MAPPING	LEVEL	JUSTIFICATION
P1-PO1	3	Applied core engineering knowledge in embedded systems
P1-PO2	2	Performed basic analysis to determine system requirements.
P1-PO3	2	Designed functional system components.
P1-PO6	2	Considered safety aspects during system development.
P1-PO12	2	Independently learned new tools/technologies relevant to ESP32.
P2-PO1	3	Applied knowledge of components and systems.
P2-PO3	2	PAnalyzed alternatives for selecting suitable components.
P2-PO6	1	Considered health, safety, and contextual implications.
P2-PO7	2	Reduce paper waste by avoiding bills
P2-PO10	3	Clearly articulated reasoning in presentations and reports.
P2-PO11	2	Used structured methodology for planning and execution
P2-PO12	2	Learned new methods and design reasoning independently.
P3-PO1	3	Applied technical knowledge to optimize the system.
P3-PO2	3	Analyzed performance data to enhance accuracy.
P3-PO3	1	Made slight design adjustments to improve the system.
P3-PO7	2	Reduce paper waste by avoiding bills
P3-PO10	3	Presented changes and results with clarity.
P3-PO11	2	Followed a planned approach for iterative improvements.
P3-PO12	2	Learned new methods during system tuning.
P4-PO8	2	Made slight design adjustments to improve the system.
P4-PO9	3	Worked effectively as a team member in integration tasks.
P4-PO10	3	Communicated integration challenges and outcomes clearly.
P4-PO11	2	Managed tasks collaboratively in the integration phase.
P4-PO12	2	Gained new integration skills through collaboration..
P5-PO8	2	Considered environmental/contextual factors in integration..
P5-PO9	3	Participated in team documentation and knowledge sharing.
P5-PO10	3	Created clear, professional documentation and presentations.
P5-PO11	2	Managed documentation responsibilities effectively.
P5-PO12	2	Improved technical writing through documentation.

Mini Project Outcome - PSO Mapping

	PSO 1	**PSO 2**	**PSO 3**
P1	3	2	2
P2	3	2	2
P3	3	1	2
P4	2	1	3
P5	2	2	2

Justifications for Mini Project Outcome - PS0 Mapping

MAPPING	**LEVEL**	**JUSTIFICATION**
P1-PSO1	3	Demonstrated strong skills in designing and testing
P1-PSO2	2	Applied basic EDA tools (e.g., Arduino IDE)
P1-PSO3	2	Showed responsibility in system decisions
P2-PSO1	3	Deep understanding of circuit design principles
P2-PSO2	2	Used appropriate simulations for selecting components
P2-PSO3	2	Showed professionalism and learning attitude
P3-PSO1	3	Applied hands-on circuit/system debugging
P3-PSO2	1	Minimal but essential usage of tools in calibration
P3-PSO3	2	Reflected a learning-oriented mindset and responsibility
P4-PSO1	2	Participated in integration of hardware/software elements
P4-PSO2	1	Limited but present tool usage for integration testing.
P4-PSO3	3	Strong demonstration of professional ethics and teamwork
P5-PSO1	2	Supported technical understanding through documentation
P5-PSO2	2	Reflected knowledge of tools through documentation
P5-PSO3	2	Consistent professionalism, communication, and growth mindset