CHAPTER 1

INTRODUCTION

The Smart Trolley for Supermarkets introduces a cutting-edge solution to revolutionize the shopping experience by leveraging RFID technology for automatic item tracking. This innovative approach enhances convenience and efficiency for both customers and retailers, transforming traditional shopping processes.

In conventional supermarkets, manual scanning of items at checkout can be time-consuming and prone to errors. The Smart Trolley eliminates these inefficiencies by automatically tracking items as they are added or removed from the trolley using RFID tags. Each item is tagged with RFID technology, enabling seamless monitoring throughout the shopping journey.

Integrated sensors within the trolley precisely measure the weight of items, ensuring accurate billing without the need for manual scanning at checkout. This streamlined process not only saves time for customers but also reduces checkout queues and enhances overall convenience.

Real-time inventory updates provided by the Smart Trolley enable retailers to efficiently manage stock levels and improve inventory control. With instant visibility into product availability, retailers can restock shelves promptly, minimizing out-of-stock situations and enhancing customer satisfaction.

The Smart Trolley elevates the shopping experience by providing customers with a frictionless journey through the aisles. With the ability to navigate effortlessly and make purchases with ease, customers enjoy a seamless and efficient shopping experience.

In summary, the Smart Trolley for Supermarkets represents a significant advancement in retail technology, enhancing efficiency, reducing checkout queues, and elevating the overall shopping experience. By leveraging RFID technology and integrated sensors, this innovation streamlines the shopping process, saving time and enhancing convenience for customers while empowering retailers with real-time inventory management capabilities.

Fig 1.1: Smart trolley

1.1: Problem Statement:

In the retail industry, traditional shopping experiences are often hindered by inefficient checkout processes and manual inventory management, leading to longer wait times and potential errors in billing. Supermarkets face the challenge of enhancing customer satisfaction while optimizing operational efficiency. To address these challenges, there is a pressing need for a Smart Trolley solution that revolutionizes the shopping experience through the integration of RFID technology and advanced sensors.

Current shopping practices rely heavily on manual scanning of items at checkout counters, which can be time-consuming and prone to errors. Additionally, traditional inventory management methods lack real-time updates, making it challenging for retailers to maintain accurate stock levels and efficiently manage inventory.

The introduction of a Smart Trolley for supermarkets seeks to address these challenges by leveraging RFID technology for automatic item tracking and integrated sensors for precise weight measurement. Each item in the supermarket is tagged with an RFID tag, allowing for seamless monitoring of additions and removals from the trolley. The integrated sensors precisely measure the weight of items, ensuring accurate billing without the need for manual scanning at checkout.

The primary objective of this project is to develop a Smart Trolley system that streamlines the shopping process, saves time, and enhances convenience for customers. By automating item tracking and billing processes, the Smart Trolley eliminates the need for manual scanning, reducing checkout queues and improving the overall shopping experience.

Furthermore, with real-time inventory updates, retailers can efficiently manage stock levels, minimize out-of-stock situations, and optimize inventory control. By providing retailers with accurate and up-to-date inventory information, the Smart Trolley system enables proactive decision-making and enhances operational efficiency.

In summary, the Smart Trolley for supermarkets aims to revolutionize the shopping experience by leveraging RFID technology and integrated sensors to automate item tracking and billing processes. This innovation enhances efficiency, reduces checkout queues, and elevates the overall shopping experience for both customers and retailers.

1.2: Problem Scope:

The implementation of a Smart Trolley System for supermarkets addresses several challenges and deficiencies within traditional shopping experiences, aiming to enhance efficiency, convenience, and customer satisfaction.

Manual Item Tracking and Billing:

Traditional shopping carts rely on manual item tracking and scanning at checkout counters, leading to inefficiencies and delays in the billing process.

- Human errors during manual tracking and scanning may result in inaccurate billing and discrepancies in the final purchase total.
- Manual scanning contributes to longer checkout queues and increased waiting times for customers.

Limited Inventory Management:

Existing systems lack real-time monitoring of inventory levels within shopping carts, hindering retailers' ability to manage stock levels effectively.

- Inaccurate inventory tracking may lead to stockouts or overstocking of items, impacting sales and operational efficiency.
- Without real-time inventory updates, retailers struggle to make informed decisions regarding restocking and inventory control.

Friction in Shopping Experience:

Customers experience friction in the shopping process due to manual item tracking and checkout procedures.

- Cumbersome scanning processes and checkout queues detract from the overall shopping experience, leading to customer dissatisfaction.
- Lack of convenience and efficiency in traditional shopping methods diminishes customer loyalty and retention.

Operational Inefficiencies:

Manual item tracking and checkout processes contribute to operational inefficiencies for both retailers and customers.

- Retailers allocate resources to manual scanning and checkout procedures, leading to increased labor costs and reduced operational productivity.
- Customers spend more time navigating aisles and waiting in checkout lines, impacting their overall shopping experience and satisfaction.

Addressing these challenges within the defined problem scope requires the development and implementation of a comprehensive Smart Trolley System. This system leverages RFID technology, integrated sensors, and real-time data processing to streamline the shopping process and enhance efficiency for both retailers and customers.

By automatically tracking items within the trolley, the Smart Trolley System eliminates the need for manual scanning at checkout counters, reducing checkout times and improving overall operational efficiency. Additionally, real-time inventory updates enable retailers to optimize stock levels and improve inventory control, leading to enhanced profitability and customer satisfaction.

Ultimately, the Smart Trolley System aims to revolutionize the shopping experience by offering seamless item tracking, accurate billing, and enhanced convenience for customers, while also providing retailers with the tools they need to optimize operations and drive business success in the competitive retail landscape.

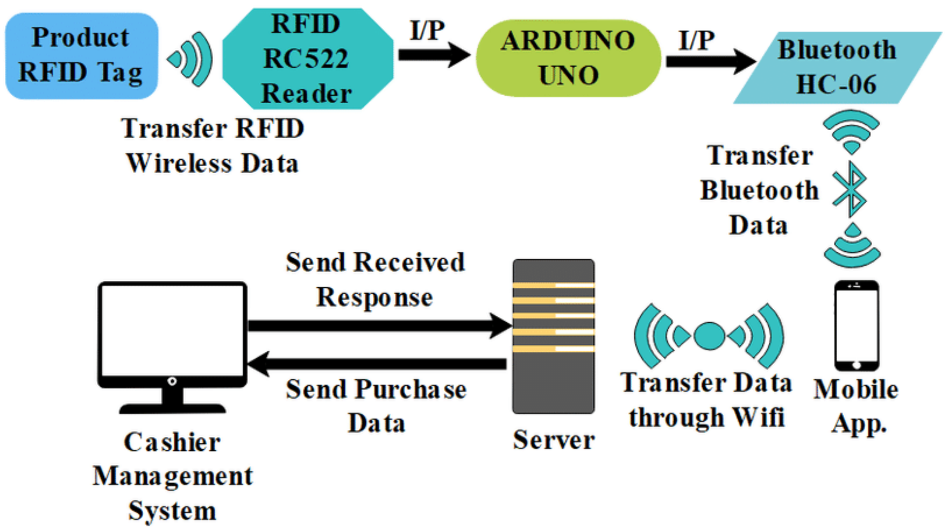


Figure 1.2: Technology behind Smart Trolley

1.3: Advantages:

The implementation of a Smart Trolley for supermarkets introduces numerous advantages, revolutionizing the shopping experience for customers and streamlining operations for retailers. Key advantages include:

Seamless Item Tracking:

RFID technology enables automatic tracking of items added to or removed from the trolley, eliminating the need for manual scanning.

Accurate Billing:

Integrated sensors precisely measure the weight of items in the trolley, ensuring accurate billing at checkout without the need for manual scanning.

Time-saving Convenience:

The innovation streamlines the shopping process, saving time for customers by eliminating the need to scan each item individually.

Real-time Inventory Updates:

Retailers benefit from real-time inventory updates, allowing them to efficiently manage stock levels and improve inventory control.

Frictionless Shopping Experience:

Customers enjoy a frictionless experience as they navigate aisles and make purchases with ease, enhancing overall satisfaction.

Reduced Checkout Queues:

By eliminating the need for manual scanning, the smart trolley reduces checkout queues, leading to faster and more efficient transactions.

Enhanced Efficiency:

The smart trolley enhances operational efficiency for both customers and retailers, streamlining the shopping process and reducing wait times.

Improved Customer Satisfaction:

With its convenience and time-saving features, the smart trolley improves customer satisfaction and loyalty, leading to repeat business.

Data-driven Insights:

Retailers can gain valuable insights into customer purchasing behavior and preferences through data collected by the smart trolley, enabling targeted marketing strategies and inventory management.

Environmental Benefits:

By reducing the need for paper receipts and manual scanning, the smart trolley contributes to environmental sustainability by minimizing paper waste and energy consumption.

Overall, the implementation of a Smart Trolley for supermarkets offers a range of benefits for both customers and retailers, enhancing the shopping experience, improving operational efficiency, and promoting environmental sustainability.

1.4 Proposed Solution:

The proposed solution for optimizing the traditional shopping experience with a Smart Trolley involves the integration of RFID technology and advanced sensors to create a seamless and efficient shopping process. Key components of the solution include:

1. RFID Technology:

Each item in the supermarket is tagged with RFID (Radio Frequency Identification) tags, enabling automatic item tracking as they are added or removed from the trolley.
RFID technology allows for quick and accurate identification of items, eliminating the need for manual scanning at checkout.

2. Integrated Sensors:

The smart trolley is equipped with integrated sensors, including weight sensors, to precisely measure the weight of items placed inside.
These sensors ensure accurate billing and provide real-time feedback on the contents of the trolley, enhancing convenience and efficiency for customers.

3. LCD Screen:

An LCD screen integrated into the trolley displays real-time information, including a list of items currently in the trolley, total cost, and promotions or discounts available.
The screen enhances the shopping experience by providing customers with instant feedback and relevant information as they shop.

4. Real-time Inventory Updates:
The RFID-enabled smart trolley continuously updates the inventory system with real-time data on items added or removed from the trolley.
Retailers can efficiently manage stock levels and improve inventory control, ensuring that shelves are adequately stocked and reducing the likelihood of out-of-stock situations.

5. Frictionless Shopping Experience:
Customers enjoy a frictionless shopping experience as they navigate the supermarket aisles with the RFID-enabled smart trolley.
With automatic item tracking and accurate billing, customers can quickly and easily complete their shopping without the need for manual scanning or checkout queues.

6. Enhanced Efficiency and Convenience:
The smart trolley streamlines the shopping process, saving time and effort for both customers and retailers.
By automating tasks such as item tracking and billing, the smart trolley reduces checkout queues and elevates the overall shopping experience.

By implementing this solution, supermarkets can leverage RFID technology and advanced sensors to optimize the shopping experience, enhance efficiency, and improve customer satisfaction. The smart trolley represents a significant advancement in retail technology, offering a seamless and convenient way for customers to shop while providing retailers with valuable insights into inventory management and customer behavior.

1.5 Aim and Objectives

Aim:

The aim of developing the smart trolley for supermarkets is to redefine the shopping experience by leveraging RFID technology and integrated sensors to streamline the process of item tracking, billing, and inventory management. This innovative solution aims to enhance convenience for customers and efficiency for retailers by automating item tracking, ensuring accurate billing, and providing real-time inventory updates. The primary objectives include creating a frictionless shopping experience for customers, reducing checkout queues, optimizing inventory control, and improving overall operational efficiency in supermarkets. By achieving these goals, the smart trolley aims to elevate the shopping experience, save time for both customers and retailers, and enhance convenience and satisfaction in the retail environment.

Objectives:

Automatic Item Tracking: Develop a smart trolley system equipped with RFID technology to automatically track items as they are added or removed from the trolley. Each item will be tagged with an RFID tag for seamless monitoring throughout the shopping process.

Precise Weight Measurement: Integrate sensors into the trolley to precisely measure the weight of items placed within it. This feature ensures accurate billing at checkout without the need for manual scanning, enhancing convenience for both customers and store staff.

Streamlined Shopping Experience: Redefine the shopping experience by streamlining the process through the use of smart trolley technology. Customers will enjoy a frictionless experience as they navigate aisles, add items to their trolley, and make purchases with ease.

Real-Time Inventory Updates: Enable real-time inventory updates through the smart trolley system. By continuously monitoring item additions and removals, retailers can efficiently manage stock levels, reduce out-of-stock situations, and improve overall inventory control.

Enhanced Efficiency: Reduce checkout queues and waiting times by implementing the smart trolley system. With automatic item tracking and precise weight measurement, the checkout process becomes faster and more efficient, enhancing the overall shopping experience for customers.

Improved Customer Convenience: Enhance convenience for customers by providing real-time updates on their purchases via an integrated LCD screen on the trolley. Customers can easily view their shopping list, total bill, and promotional offers as they shop.

Optimized Inventory Management: Empower retailers with valuable insights into customer shopping behavior and inventory levels. By analyzing data collected from the smart trolley system, retailers can make informed decisions to optimize inventory management and enhance sales performance.

Seamless Integration: Ensure seamless integration of RFID technology and LCD screen into the design of the smart trolley. The components should work harmoniously to deliver a user-friendly and reliable shopping experience for both customers and retailers.

By achieving these objectives, the smart trolley for supermarkets aims to revolutionize the shopping experience, offering customers convenience, efficiency, and a seamless checkout process, while empowering retailers with improved inventory management capabilities and enhanced customer satisfaction.

CHAPTER 2

Literature Survey

The advent of Smart Trolleys for supermarkets has ushered in a new era of shopping convenience, as evidenced by the burgeoning literature in this field. Researchers have delved into the transformative impact of RFID technology in redefining the shopping experience, particularly in terms of automatic item tracking and accurate billing. By equipping each item with RFID tags, supermarkets can seamlessly monitor additions and removals from trolleys, streamlining the checkout process and enhancing customer satisfaction.

Integrated sensors play a pivotal role in ensuring the accuracy of billing by precisely measuring the weight of items in the trolley. This innovation not only eliminates the need for manual scanning at checkout but also minimizes errors, thus enhancing efficiency and reducing checkout queues. The literature highlights the significant time savings and convenience afforded to customers through this frictionless shopping experience, ultimately elevating the overall shopping experience.

Real-time inventory updates facilitated by Smart Trolleys enable retailers to efficiently manage stock levels and improve inventory control. By harnessing RFID technology and integrated sensors, supermarkets can optimize their inventory management processes, leading to reduced stockouts, minimized wastage, and increased profitability. Customers benefit from the availability of products and enjoy a seamless shopping experience as they navigate aisles and make purchases with ease.

The literature survey emphasizes the multifaceted advantages of Smart Trolleys for both retailers and customers. From enhancing efficiency and reducing checkout queues to improving inventory control and elevating the overall shopping experience, Smart Trolleys represent a significant advancement in the retail sector. The exploration of RFID technology, integrated sensors, and real-time inventory management underscores a concerted effort to innovate and enhance the retail landscape, ultimately shaping the future of shopping.

CHAPTER 3

Methodology

The methodology for implementing a Smart Trolley for supermarkets with RFID technology follows a systematic approach tailored to the unique demands of modern retail environments. It begins with a comprehensive needs assessment involving stakeholders such as supermarket management and customers. Through this assessment, critical requirements and operational challenges associated with traditional shopping processes are identified, including the need for efficient inventory management, streamlined checkout procedures, and an enhanced customer experience.

Subsequently, both functional and technical requirements for the Smart Trolley system are clearly defined. These requirements encompass automatic item tracking using RFID technology, integration with supermarket inventory systems, and real-time data updates to ensure accurate billing and inventory management.

The selection of components for the Smart Trolley system is conducted meticulously, taking into account factors such as compatibility, reliability, and cost-effectiveness. Key components include RFID tags and readers for automatic item tracking, as well as an LCD screen for displaying item information and transaction details to customers.

System development focuses on seamlessly integrating RFID technology for automatic item tracking and weight sensors for precise billing. The development process also entails designing a user-friendly interface on the LCD screen to display item details, prices, and total costs. Real-time inventory updates are implemented to enable efficient stock management and enhance overall operational efficiency.

Security implementation is prioritized to protect customer data and ensure the integrity of transactions. Encryption protocols and authentication mechanisms are incorporated to safeguard sensitive information transmitted between the Smart Trolley and supermarket systems.

Integration with existing supermarket infrastructure, including inventory management systems and checkout terminals, is carefully addressed to ensure interoperability and seamless data exchange.

Once developed, the Smart Trolley system undergoes rigorous testing to verify functionality, reliability, and accuracy. Any issues or discrepancies identified during testing are addressed promptly to ensure optimal system performance.

Training programs are provided to supermarket staff on the operation and maintenance of the Smart Trolley system. Deployment is carried out in phases, starting with pilot testing in select supermarket locations before full-scale implementation. Continuous monitoring and feedback gathering enable ongoing optimization and improvement of the Smart Trolley system to meet evolving supermarket needs and customer expectations.

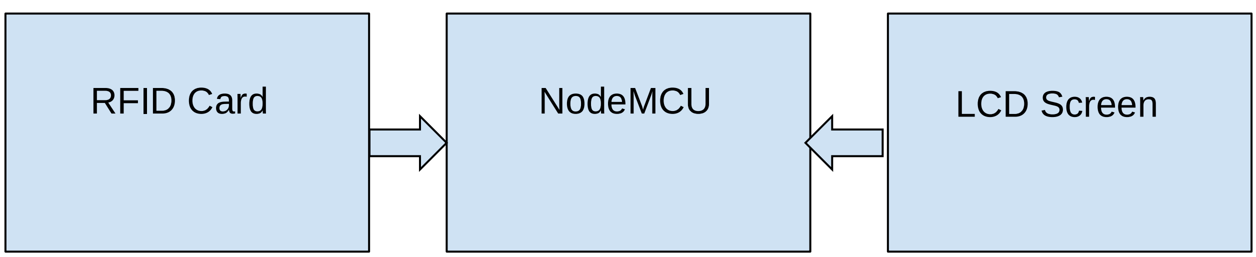


Figure 3.1: Block Diagram of the Smart Trolley

3.1 NodeMCU (ESP8266)

The NodeMCU ESP8266 is a powerful and versatile platform designed for Internet of Things (IoT) development. It is a cost-effective Wi-Fi microchip known for its capability to enable wireless communication in IoT applications. NodeMCU, on the other hand, is an open-source firmware and development kit that simplifies the process of prototyping and programming the ESP8266, built-in Wi-Fi connectivity, the NodeMCU ESP8266 allows devices to connect to the internet wirelessly, making it suitable for a wide range of IoT projects. One notable feature is its support for the Lua scripting language, providing a high-level programming environment for developers. Additionally, it is compatible with the Arduino IDE, allowing those familiar with Arduino to use the NodeMCU platform. Equipped with General Purpose Input/Output (GPIO) pins, the ESP8266 facilitates interfacing with various electronic components, making it ideal for applications such as home automation and sensor networks. It has garnered significant community support, resulting in an extensive collection of libraries and documentation, making it a popular choice for rapid IoT prototyping and development.

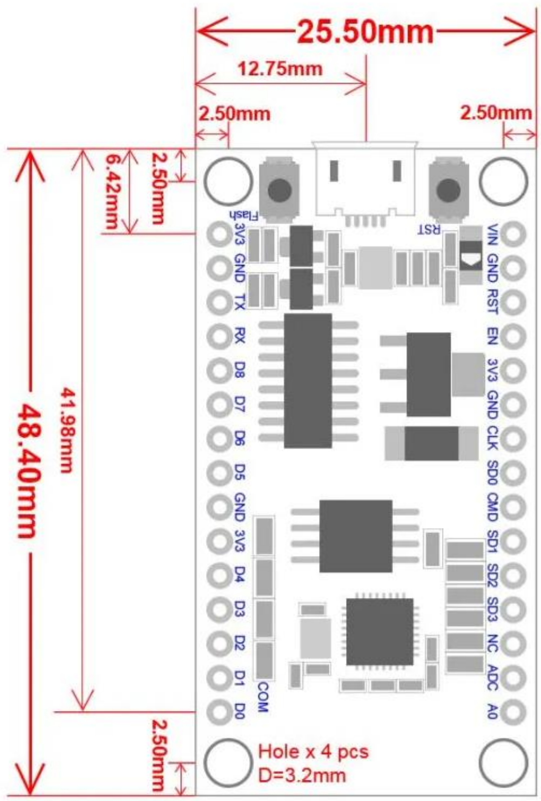


Figure 3.2 NodeMCU 2D View

NodeMCU Specification:

The NodeMCU development board is based on the ESP8266 microcontroller, and different versions of NodeMCU boards may have slight variations in specifications. As of my knowledge cutoff in January 2022, here are the general specifications for the NodeMCU ESP8266 development board:

1. Microcontroller: ESP8266 Wi-Fi microcontroller with 32-bit architecture.

2. Processor: Tensilica L106 32-bit microcontroller.

3. Clock Frequency: Typically operates at 80 MHz.

4. Flash Memory:

- Built-in Flash memory for program storage.
- Common configurations include 4MB or 16MB of Flash memory.

5. RAM: Typically equipped with 80 KB of RAM.

6. Wireless Connectivity:

- Integrated Wi-Fi (802.11 b/g/n) for wireless communication.
- Supports Station, SoftAP, and SoftAP + Station modes.

7. GPIO Pins: Multiple General Purpose Input/Output (GPIO) pins for interfacing with sensors, actuators, and other electronic components.

8. Analog Pins: Analog-to-digital converter (ADC) pins for reading analog sensor values.

9. USB-to-Serial Converter: Built-in USB-to-Serial converter for programming and debugging.

10. Operating Voltage: Typically operates at 3.3V (Note: It is crucial to connect external components accordingly to avoid damage).

11. Programming Interface: Programmable using the Arduino IDE, Lua scripting language, or other compatible frameworks.

12. Voltage Regulator: Onboard voltage regulator for stable operation.

13. Reset Button: Reset button for restarting the board.

14. Dimensions: Standard NodeMCU boards often have dimensions around 49mm x 24mm.

15. Power Consumption: Low power consumption, making it suitable for battery-operated applications.

16. Community Support: Active community support with extensive documentation and libraries.

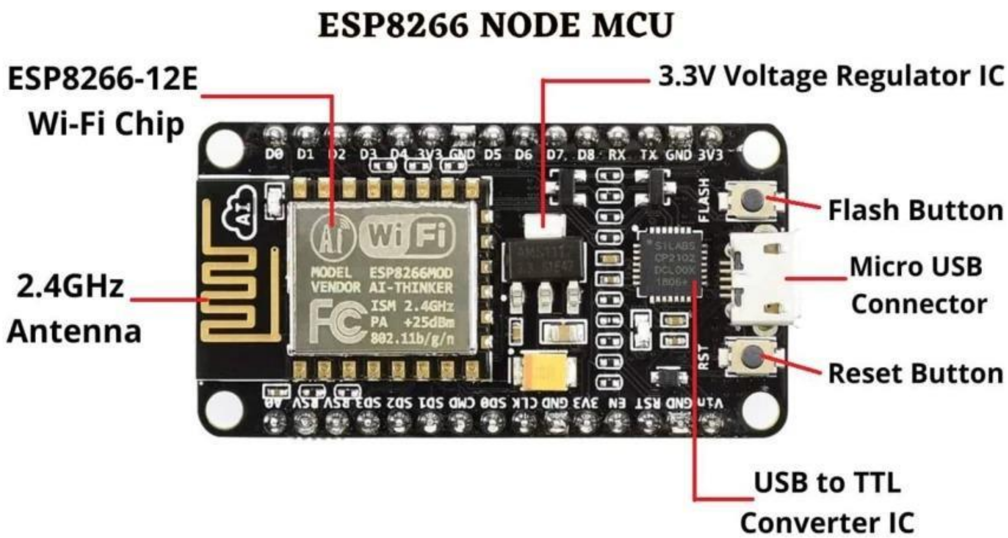


Figure 3.3: NodeMCU Parts

The NodeMCU ESP8266 development board typically has GPIO (General Purpose Input/Output) pins that can be used for various purposes, including interfacing with sensors, actuators, and other electronic components. Below is a common pinout configuration for the NodeMCU development board

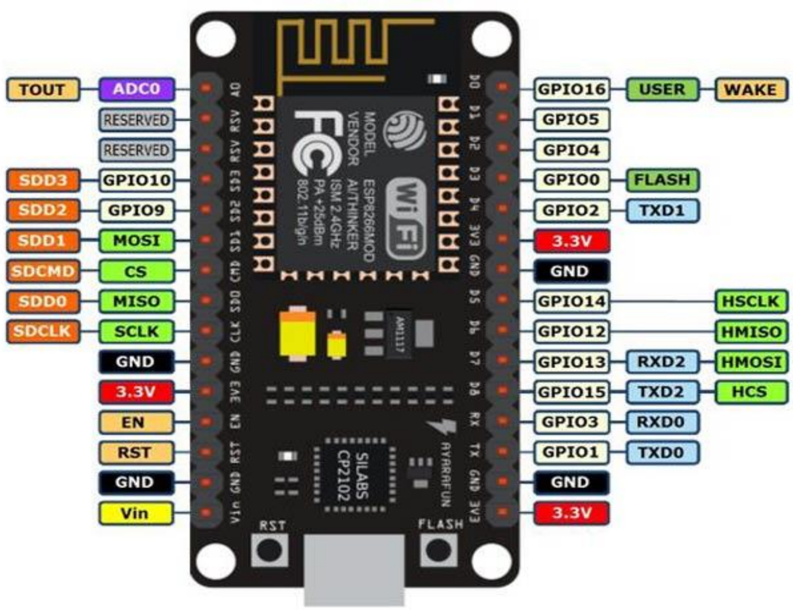


Figure 3.4: NodeMCU ESP8266 Pinout

ADC | A0 | GPIO16

EN | Enable | GPIO14

D0 | GPIO16 | GPIO12

D1	GPIO5	GPIO13
D2	GPIO4	GPIO15
D3	GPIO0	GPIO2
D4	GPIO2	GPIO9
D5	GPIO14	GPIO10
D6	GPIO12	GPIO3
D7	GPIO13	GPIO1
D8	GPIO15	TX (GPIO1)
D9	GPIO3 (RX)	RX (GPIO3)
D10	GPIO1 (TX)	D11 (MOSI)
D11	MOSI	D12 (MISO)
D12	MISO	D13 (SCK

ADC: Analog-to-Digital Converter pin for reading analog sensor values.
EN (Enable): Enable pin.
D0-D8: Digital GPIO pins.
D9 (RX) and D10 (TX): Serial communication pins for programming and debugging.
D11 (MOSI), D12 (MISO), D13 (SCK): Pins used for SPI communication.
D14 (SDA) and D15 (SCL): Pins used for I2C communication.
It's important to note that GPIO pins labeled as "D" (Digital) are typically used for general-purpose digital input/output. Additionally, GPIO pins labeled as "A" (Analog) can be used as analog inputs with the ADC. GPIO pins 6, 7, 8, 9, 10, and 11 have additional functions, so it's advised to refer to the specific NodeMCU documentation for detailed information on pin functionality and capabilities.

3.2 Arduino software:

Arduino microcontrollers are programmed using the Arduino IDE (Integrated Development Environment). Arduino programs, called “sketches”, are written in a programming language similar to C and C++. Every sketch must have a setup () function (executed just once) followed by a loop () function (potentially executed many times); add “comments” to code to make it easier to read. Many sensors and other hardware devices come with prewritten software line for sample code, libraries (of functions). Libraries are a collection of code that makes it easy for you to connect to a sensor, display, module, etc. For example, the built-in Liquid Crystal library makes it easy to talk to character LCD displays. There are hundreds of additional libraries available on the internet for download.

Fig 3.5: Arduino IDE

3.3 Liquid Crystal Display (LCD):

Figure 3.6: Liquid Crystal Display

A LCD is a tool used for visual display of the output. The liquid -crystal display has the distinct advantage of having low power consumption than the LED. It is typically of the order of microwatts for the display in comparison to the some order of milli watts for LEDs. Low power consumption requirement has made it compatible with MOS integrated logic circuit. Its other advantages are its low cost, and good contrast. The main drawbacks of LCDs are additional requirement of light source, a limited temperature range of operation (between 0 and 60° C), low reliability, short operating life, poor visibility in low ambient lighting.

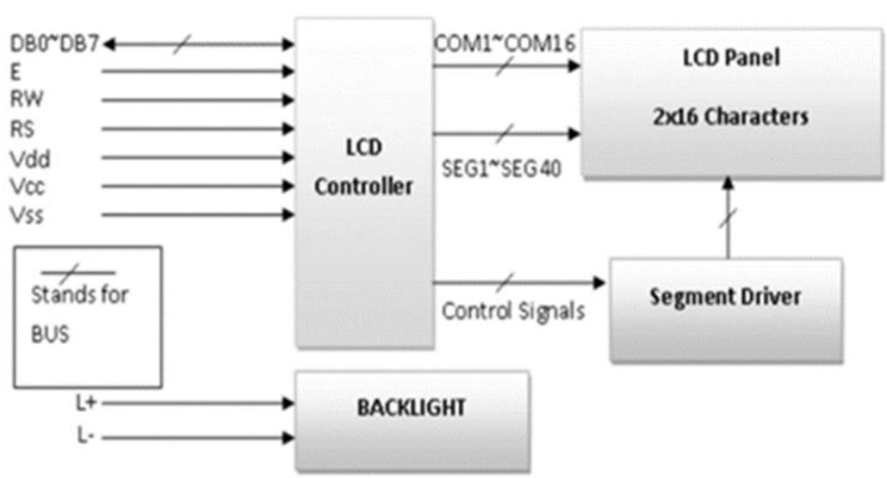


Figure 3.7: Block diagram of LCD

The principle behind the LCDs is that when an electrical current is applied to the liquid crystal molecule, the molecule tends to untwist. This causes the angle of light which is passing through the molecule of the polarized glass and also causes a change in the angle of the top polarizing filter. As a result, a little light is allowed to pass the polarized glass through a particular area of the LCD.

Thus that particular area will become dark compared to others. The LCD works on the principle of blocking light. While constructing the LCDs, a reflected mirror is arranged at the back. An electrode plane is made of indium-tin-oxide which is kept on top and a polarized glass with a polarizing film is also added on the bottom of the device. The complete region of the LCD has to be enclosed by a common electrode and above it should be the liquid crystal matter.

Next comes the second piece of glass with an electrode in the form of the rectangle on the bottom and, on top, another polarizing film. It must be considered that both the pieces are kept at the right angles. When there is no current, the light passes through the front of the LCD and it will be reflected by the mirror and bounced back. As the electrode is connected to a battery the current from it will cause the liquid crystals between the common-plane electrode and the electrode shaped like a rectangle to untwist. Thus, the light is blocked from passing through. That particular rectangular area appears blank.

3.3.1 LCD Features:

- Operating Voltage is 4.7V to 5.3V.
- Current consumption is 1mA without backlight.
- Alphanumeric LCD display module, meaning can display alphabets and numbers.
- Consists of two rows and each row can print 16 characters.
- Each character is built by a 5×8 pixel box.
- Can work in both 8-bit and 4-bit mode.

● It can also display any custom generated characters.

● Available in Green and Blue Backlight.

3.3.2 Basic Structure of an LCD:

A liquid crystal cell consists of a thin layer (about 10 um) of a liquid crystal sandwiched between two glass sheets with transparent electrodes deposited on their inside faces. With both glass sheets transparent, the cell is known as a transmitting type cell. When one glass is transparent and the other has a reflective coating, the cell is called reflective type. The LCD does not produce any illumination of its own. It, in fact, depends on illumination falling on it from an external source for its visual effect.

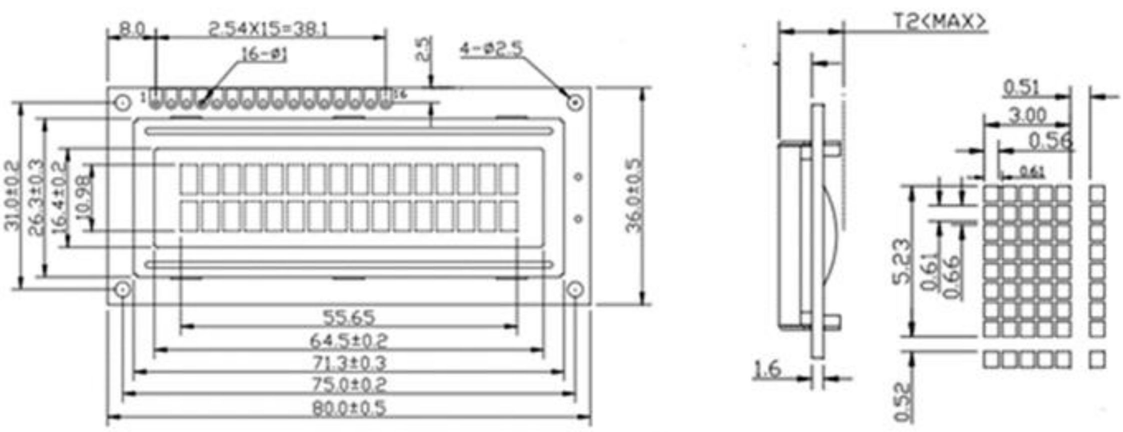


Figure 3.8: 2D of an LCD

3.4 RFID:

Radio frequency identification (RFID) is a cutting-edge technology that harnesses radio waves to identify and monitor objects or people effortlessly without physical contact. This innovative system comprises three essential elements:

RFID tags, which are tiny devices that store data

RFID readers, which wirelessly communicate with the tags

A backend system, which manages and processes the collected information

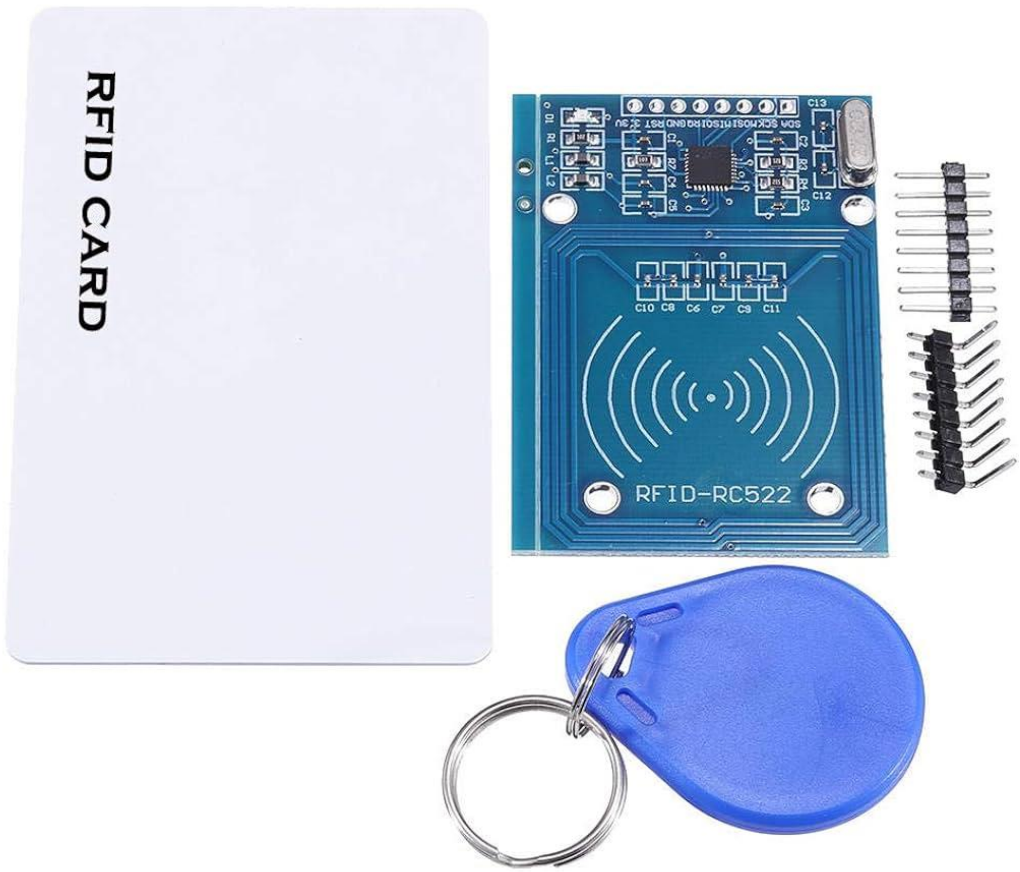


Figure 3.9 RFID

Details of flow sensor

RFID tags

Physical structure: RFID tags comprise an integrated circuit (IC) or microchip and an antenna. The IC contains non-volatile memory, which stores data and a unique identification number.

Frequency bands: RFID tags operate in different frequency bands, such as low frequency (LF: 125 kHz and 134 kHz), high frequency (HF: 13.56 MHz), and ultra-high frequency (UHF: 860-960 MHz). Each frequency band has its advantages in terms of read range, data transfer speed, and resistance to interference.

Data transfer: When energized by an RFID reader’s radio waves, the tag’s antenna receives power and activates the IC. The IC then modulates the radio waves and backscatters them to the reader, transmitting the stored data (UID and additional information).

Types of tags: RFID tags can be categorized as passive, active, or semi-passive (battery-assisted passive). Passive tags rely on the reader’s energy for power, while active tags have their own power source (battery) to actively transmit signals. Semi-passive tags use the reader’s energy for powering the IC but have a battery for signal transmission.

RFID readers

Radio wave emission: RFID readers emit radio waves in a specific frequency band. The emitted waves serve as energy sources and communication carriers.

Antenna design: RFID antennas can be classified into linear polarized, circular polarized, and near-field antennas. The size and shape of the antenna depend on the frequency band used and the desired read range. Higher-frequency RFID systems usually have smaller antenna sizes compared to lower-frequency systems. Antennas can be designed as dipoles, loops, patches, or customized shapes depending on the application requirements.

Data communication: RFID readers communicate with tags through electromagnetic coupling. They emit continuous or pulsed radio waves to energize nearby tags and receive their responses. The reader’s transceiver demodulates and decodes the tag’s response, extracting the transmitted data.

Communication protocols: RFID readers use different communication protocols, such as EPCglobal Gen2 for UHF RFID, ISO/IEC 15693 for HF RFID, and ISO/IEC 14443 for near field communication (NFC) applications.

Features of RFID:

1. Contactless Operation: RFID systems allow data to be transmitted wirelessly between an RFID tag and a reader without the need for direct contact. This feature enables quick and convenient identification of objects or individuals without manual intervention.
2. Unique Identification: Each RFID tag contains a unique identifier, known as the Electronic Product Code (EPC) or serial number. This unique identifier allows for accurate and reliable identification of individual items, assets, or entities.
3. Data Storage Capacity: RFID tags can store varying amounts of data depending on their type and memory capacity. This data may include product information, manufacturing details, expiration dates, or other relevant information associated with the tagged item.
4. Versatility: RFID technology can be deployed in various forms, including passive, active, and semi-passive tags, as well as different frequency bands such as low frequency (LF), high frequency (HF), and ultra-high frequency (UHF). This versatility allows RFID systems to be customized to suit specific application requirements.
5. Read Range: RFID systems offer adjustable read ranges, allowing users to configure the distance at which RFID tags can be read by the reader. This flexibility enables optimization for different use cases, from short-range applications such as access control to long-range applications such as supply chain management.
6. Durability: RFID tags are available in a range of form factors and materials, including ruggedized tags designed to withstand harsh environmental conditions such as extreme temperatures, moisture, and physical impact. This durability ensures reliable performance in diverse operating environments.
7. Real-Time Tracking: RFID systems enable real-time tracking and monitoring of tagged objects or individuals as they move through a predefined space or along a supply chain. This real-time visibility enhances inventory management, asset tracking, and logistics operations, leading to improved efficiency and productivity.
8. Security: RFID technology offers various security features, including encryption and authentication protocols, to protect data integrity and prevent unauthorized access or

tampering. These security measures ensure the confidentiality and reliability of information transmitted between RFID tags and readers.

9. Integration: RFID systems can be seamlessly integrated with existing infrastructure and enterprise systems, such as inventory management software, ERP (Enterprise Resource Planning) systems, and supply chain management platforms. This integration facilitates data sharing and interoperability across different business processes and applications.

10.

Connection:

SDA: Serial Data Pin for SPI Communication.

SCK: Serial Clock Pin for SPI Communication.

MOSI: Master Out Slave In pin for SPI Communication.

MISO: Master In Slave Out pin for SPI Communication.

RQ: Interrupt Request pin of the module.

GND: Connected to the ground.

Specifications:

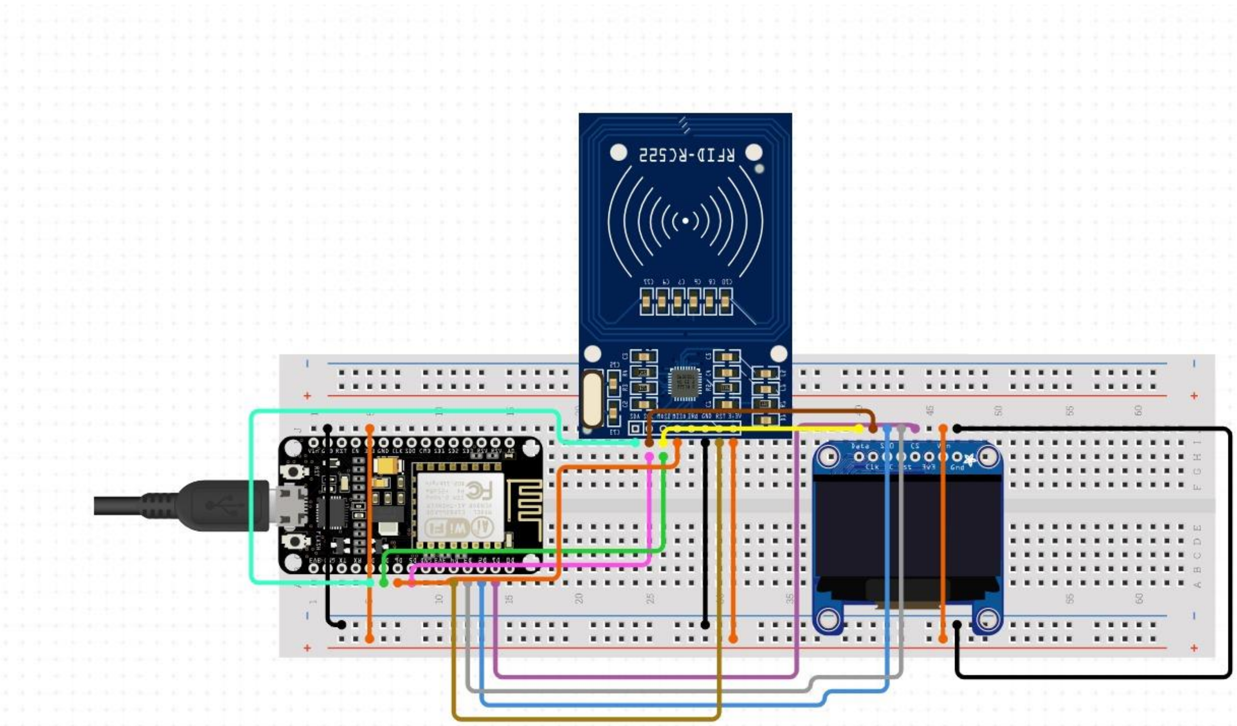
- Operating voltage: 4.8 V - 7.2 V
- Working current 13-26mA/DC 3.3V
- Idle current 10-13mA / DC 3.3V
- Sleep current <80uA
- Peak current <30mA
- Working frequency 13.56MHz

- Product phy. characteristics: size: 40mm × 60mm
- Environmental working temp.   -20-80 degrees Celsius
- Environmental storage temp. -40-85 degrees Celsius
- Relative humidity   5%-95% relative humidity
- Data transmission rateup to 10Mbit/s

CHAPTER 4

Design and Coding

4.1 Circuit Diagram



4.2 Connections :

Trolley 1 :

- RFID v3.3 to ESP8266 3.3v
- RFID RST to ESP8266 D4
- RFID GND to ESP8266 GND
- RFID MISO to ESP8266 D6
- RFID MOSI to ESP8266 D7
- RFID SCK to ESP8266 D5
- RFID SDA to ESP8266 D8

- OLED VDD to ESP8266 3.3v
- OLED GND to ESP8266 GND
- OLED SCK to ESP8266 D1
- OLED SDA to ESP8266 D2

Trolley 2 :

- RFID v3.3 to ESP8266 3.3v
- RFID RST to ESP8266 D4
- RFID GND to ESP8266 GND
- RFID MISO to ESP8266 D6
- RFID MOSI to ESP8266 D7
- RFID SCK to ESP8266 D5
- RFID SDA to ESP8266 D8
- OLED VDD to ESP8266 3.3v
- OLED GND to ESP8266 GND
- OLED SCK to ESP8266 D1
- OLED SDA to ESP8266 D2

Register office:

- RFID v3.3 to ESP8266 3.3v
- RFID RST to ESP8266 D4
- RFID GND to ESP8266 GND
- RFID MISO to ESP8266 D6
- RFID MOSI to ESP8266 D7
- RFID SCK to ESP8266 D5
- RFID SDA to ESP8266 D8
- OLED VDD to ESP8266 3.3v
- OLED GND to ESP8266 GND
- OLED SCK to ESP8266 D1
- OLED SDA to ESP8266 D2

Code:

Trolley 1 :

```
#include <SPI.h>
#include <MFRC522.h>
#include <Arduino.h>
#include <U8g2lib.h>
#include <ESP8266WiFi.h>
#include <Firebase_ESP_Client.h>
#include "addons/TokenHelper.h"
#include "addons/RTDBHelper.h"

#ifdef U8X8_HAVE_HW_SPI
#include <SPI.h>
#endif
#ifdef U8X8_HAVE_HW_I2C
#include <Wire.h>
#endif

constexpr uint8_t RST_PIN = D4;     // Configurable, see typical pin layout above
constexpr uint8_t SS_PIN = D8;     // Configurable, see typical pin layout above
MFRC522 rfid(SS_PIN, RST_PIN); // Instance of the class
MFRC522::MIFARE_Key key;

U8G2_SSD1306_128X64_NONAME_F_SW_I2C u8g2(U8G2_R0, /* clock=*/ D1, /* data=*/ D2,
/* reset=*/ U8X8_PIN_NONE);   // All Boards without Reset of the Display

#define WIFI_SSID "123456789"
#define WIFI_PASSWORD "123456789"
#define API_KEY "AIzaSyBuz9awICx9wnE44zeVrgGy_JSiI51JfcE"
#define DATABASE_URL "https://smart-billboards-using-iot-default-rtdb.firebaseio.com/"

FirebaseData fbdo;
FirebaseAuth auth;
FirebaseConfig config;
unsigned long sendDataPrevMillis = 0;
bool signupOK = false;
String intValue;

void setup() {
```

```
  Serial.begin(115200);
  SPI.begin(); // Init SPI bus
  rfid.PCD_Init(); // Init MFRC522

  u8g2.begin();
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED){
    Serial.print(".");
    delay(300);
  }
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();
  config.api_key = API_KEY;
  config.database_url = DATABASE_URL;
  if (Firebase.signUp(&config, &auth, "", "")){
    Serial.println("ok");
    signupOK = true;
  }
  else{
    Serial.printf("%s\n", config.signer.signupError.message.c_str());
  }
  config.token_status_callback = tokenStatusCallback; //see addons/TokenHelper.h
  Firebase.begin(&config, &auth);
  Firebase.reconnectWiFi(true);
}

void loop() {
  if (!rfid.PICC_IsNewCardPresent())
    return;
  if (rfid.PICC_ReadCardSerial()) {
    String tag;
    for (byte i = 0; i < 4; i++) {
      tag += rfid.uid.uidByte[i];
    }
    Serial.println("Detected Card UID: " + tag);
    // Check if the detected UID is in the list of authorized UIDs
    delay(100);
    u8g2.clearBuffer();          // clear the internal memory
    u8g2.setFont(u8g2_font_ncenB08_tr); // choose a suitable font
    u8g2.drawStr(0,10,"Verifying");  // write something to the internal memory
    u8g2.sendBuffer();
```

```
  delay(100);

  if (Firebase.ready() && signupOK && (millis() - sendDataPrevMillis > 1000 ||
sendDataPrevMillis == 0)){
  sendDataPrevMillis = millis();

  if (Firebase.RTDB.setString(&fbdo, "smarttravel/check1", tag)){
    Serial.println("PATH: " + fbdo.dataPath());
    Serial.println("TYPE: " + fbdo.dataType());
  }
  else {
    Serial.println("Failed REASON: " + fbdo.errorReason());
  }
  // transfer internal memory to the display
  delay(4000);

  if (Firebase.RTDB.getString(&fbdo, "/smarttravel/access1"))
  {
    intValue = fbdo.stringData();
    String mySubString = intValue.substring(2, 3);
    Serial.println(intValue);
    Serial.println(mySubString);
    if (mySubString == "a")
    {
      u8g2.clearBuffer();          // clear the internal memory
      u8g2.setFont(u8g2_font_ncenB08_tr); // choose a suitable font
      u8g2.drawStr(0,10,"Access Approved");  // write something to the internal memory
      u8g2.sendBuffer();          // transfer internal memory to the display
      delay(5000);
      u8g2.clearBuffer();          // clear the internal memory
      u8g2.sendBuffer();          // transfer internal memory to the display
      delay(100);
    }
    else if (mySubString == "b")
    {
      u8g2.clearBuffer();          // clear the internal memory
      u8g2.setFont(u8g2_font_ncenB08_tr); // choose a suitable font
      u8g2.drawStr(0,10,"Access Denied");  // write something to the internal memory
      u8g2.drawStr(0,40,"Please check your");  // write something to the internal memory
      u8g2.drawStr(0,50,"balance");  // write something to the internal memory
      u8g2.sendBuffer();          // transfer internal memory to the display
      delay(5000);
      u8g2.clearBuffer();          // clear the internal memory
      u8g2.sendBuffer();
```

```
      delay(100);
    }
  delay(100);
  }
  else {
    Serial.println(fbdo.errorReason());
  }
  delay(100);
  }
  rfid.PICC_HaltA();
  rfid.PCD_StopCrypto1();
  }
}
```

Trolley 2 :

```
#include <SPI.h>
#include <MFRC522.h>
#include <Arduino.h>
#include <U8g2lib.h>
#include <ESP8266WiFi.h>
#include <Firebase_ESP_Client.h>
#include "addons/TokenHelper.h"
#include "addons/RTDBHelper.h"

#ifdef U8X8_HAVE_HW_SPI
#include <SPI.h>
#endif
#ifdef U8X8_HAVE_HW_I2C
#include <Wire.h>
#endif

constexpr uint8_t RST_PIN = D4;     // Configurable, see typical pin layout above
constexpr uint8_t SS_PIN = D8;     // Configurable, see typical pin layout above
MFRC522 rfid(SS_PIN, RST_PIN); // Instance of the class
MFRC522::MIFARE_Key key;

U8G2_SSD1306_128X64_NONAME_F_SW_I2C u8g2(U8G2_R0, /* clock=*/ D1, /* data=*/ D2,
/* reset=*/ U8X8_PIN_NONE);   // All Boards without Reset of the Display

#define WIFI_SSID "123456789"
```

```
#define WIFI_PASSWORD "123456789"
#define API_KEY "AIzaSyBuz9awICx9wnE44zeVrgGy_JSiI51JfcE"
#define DATABASE_URL "https://smart-billboards-using-iot-default-rtdb.firebaseio.com/"

FirebaseData fbdo;
FirebaseAuth auth;
FirebaseConfig config;
unsigned long sendDataPrevMillis = 0;
bool signupOK = false;
String intValue;

void setup() {
  Serial.begin(115200);
  SPI.begin(); // Init SPI bus
  rfid.PCD_Init(); // Init MFRC522

  u8g2.begin();
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED){
    Serial.print(".");
    delay(300);
  }
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();
  config.api_key = API_KEY;
  config.database_url = DATABASE_URL;
  if (Firebase.signUp(&config, &auth, "", "")){
    Serial.println("ok");
    signupOK = true;
  }
  else{
    Serial.printf("%s\n", config.signer.signupError.message.c_str());
  }
  config.token_status_callback = tokenStatusCallback; //see addons/TokenHelper.h
  Firebase.begin(&config, &auth);
  Firebase.reconnectWiFi(true);
}

void loop() {
  if (!rfid.PICC_IsNewCardPresent())
    return;
```

```
  if (rfid.PICC_ReadCardSerial()) {
    String tag;
    for (byte i = 0; i < 4; i++) {
      tag += rfid.uid.uidByte[i];
    }
    Serial.println("Detected Card UID: " + tag);
    // Check if the detected UID is in the list of authorized UIDs
    delay(100);
    u8g2.clearBuffer();          // clear the internal memory
    u8g2.setFont(u8g2_font_ncenB08_tr); // choose a suitable font
    u8g2.drawStr(0,10,"Verifying");  // write something to the internal memory
    u8g2.sendBuffer();
    delay(100);

    if (Firebase.ready() && signupOK && (millis() - sendDataPrevMillis > 1000 ||
sendDataPrevMillis == 0)){
    sendDataPrevMillis = millis();

    if (Firebase.RTDB.setString(&fbdo, "smarttravel/check2", tag)){
      Serial.println("PATH: " + fbdo.dataPath());
      Serial.println("TYPE: " + fbdo.dataType());
    }
    else {
      Serial.println("Failed REASON: " + fbdo.errorReason());
    }
    // transfer internal memory to the display
    delay(4000);

    if (Firebase.RTDB.getString(&fbdo, "/smarttravel/access2"))
    {
      intValue = fbdo.stringData();
      String mySubString = intValue.substring(2, 3);
      Serial.println(intValue);
      Serial.println(mySubString);
      if (mySubString == "a")
      {
        u8g2.clearBuffer();          // clear the internal memory
        u8g2.setFont(u8g2_font_ncenB08_tr); // choose a suitable font
        u8g2.drawStr(0,10,"Access Approved");  // write something to the internal memory
        u8g2.sendBuffer();          // transfer internal memory to the display
        delay(5000);
        u8g2.clearBuffer();          // clear the internal memory
        u8g2.sendBuffer();          // transfer internal memory to the display
        delay(100);
```

```
      }
      else if (mySubString == "b")
      {
        u8g2.clearBuffer();          // clear the internal memory
        u8g2.setFont(u8g2_font_ncenB08_tr); // choose a suitable font
        u8g2.drawStr(0,10,"Access Denied");  // write something to the internal memory
        u8g2.drawStr(0,40,"Please check your");  // write something to the internal memory
        u8g2.drawStr(0,50,"balance");  // write something to the internal memory
        u8g2.sendBuffer();          // transfer internal memory to the display
        delay(5000);
        u8g2.clearBuffer();          // clear the internal memory
        u8g2.sendBuffer();
        delay(100);
      }
    delay(100);
    }
    else {
      Serial.println(fbdo.errorReason());
    }
    delay(100);
    }
    rfid.PICC_HaltA();
    rfid.PCD_StopCrypto1();
    }
  }
```

Register office:

```
#include <SPI.h>
#include <MFRC522.h>
#include <Arduino.h>
#include <U8g2lib.h>
#include <ESP8266WiFi.h>
#include <Firebase_ESP_Client.h>
#include "addons/TokenHelper.h"
#include "addons/RTDBHelper.h"

#ifdef U8X8_HAVE_HW_SPI
#include <SPI.h>
#endif
#ifdef U8X8_HAVE_HW_I2C
#include <Wire.h>
#endif
```

```
constexpr uint8_t RST_PIN = D4;     // Configurable, see typical pin layout above
constexpr uint8_t SS_PIN = D8;     // Configurable, see typical pin layout above
MFRC522 rfid(SS_PIN, RST_PIN); // Instance of the class
MFRC522::MIFARE_Key key;

const int buzzerPin = D3;  // Replace with the actual pin connected to the buzzer
U8G2_SSD1306_128X64_NONAME_F_SW_I2C u8g2(U8G2_R0, /* clock=*/ D1, /* data=*/ D2,
/* reset=*/ U8X8_PIN_NONE);   // All Boards without Reset of the Display

#define WIFI_SSID "123456789"
#define WIFI_PASSWORD "123456789"
#define API_KEY "AIzaSyBuz9awICx9wnE44zeVrgGy_JSiI51JfcE"
#define DATABASE_URL "https://smart-billboards-using-iot-default-rtdb.firebaseio.com"

FirebaseData fbdo;
FirebaseAuth auth;
FirebaseConfig config;
unsigned long sendDataPrevMillis = 0;
bool signupOK = false;
String intValue;

void setup() {
  Serial.begin(115200);
  SPI.begin(); // Init SPI bus
  rfid.PCD_Init(); // Init MFRC522

  pinMode(buzzerPin, OUTPUT);
  u8g2.begin();
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED){
    Serial.print(".");
    delay(300);
  }
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();
  config.api_key = API_KEY;
  config.database_url = DATABASE_URL;
  if (Firebase.signUp(&config, &auth, "", "")){
    Serial.println("ok");
    signupOK = true;
```

```
  }
  else{
    Serial.printf("%s\n", config.signer.signupError.message.c_str());
  }
  config.token_status_callback = tokenStatusCallback; //see addons/TokenHelper.h
  Firebase.begin(&config, &auth);
  Firebase.reconnectWiFi(true);
}

void loop() {
  if (!rfid.PICC_IsNewCardPresent())
    return;
  if (rfid.PICC_ReadCardSerial()) {
    String tag;
    for (byte i = 0; i < 4; i++) {
      tag += rfid.uid.uidByte[i];
    }
    Serial.println("Detected Card UID: " + tag);
    // Check if the detected UID is in the list of authorized UIDs
    delay(100);
    u8g2.clearBuffer();          // clear the internal memory
    u8g2.setFont(u8g2_font_ncenB08_tr); // choose a suitable font
    u8g2.drawStr(0,10,"Welcome");  // write something to the internal memory
    u8g2.drawStr(0,30,"process in App");  // write something to the internal memory
    u8g2.sendBuffer();
    delay(100);

    if (Firebase.ready() && signupOK && (millis() - sendDataPrevMillis > 1000 ||
sendDataPrevMillis == 0)){
    sendDataPrevMillis = millis();

    if (Firebase.RTDB.setString(&fbdo, "smarttravel/add", tag)){
      Serial.println("PATH: " + fbdo.dataPath());
      Serial.println("TYPE: " + fbdo.dataType());
    }
    else {
      Serial.println("Failed REASON: " + fbdo.errorReason());
    }
    // transfer internal memory to the display
    delay(4000);

    if (Firebase.RTDB.getString(&fbdo, "/smarttravel/result"))
    {
      intValue = fbdo.stringData();
```

```
    String mySubString = intValue.substring(2, 3);
    Serial.println(intValue);
    Serial.println(mySubString);
    if (mySubString == "a")
    {
      u8g2.clearBuffer();          // clear the internal memory
      u8g2.setFont(u8g2_font_ncenB08_tr); // choose a suitable font
      u8g2.drawStr(0,10,"User Added");  // write something to the internal memory
      u8g2.sendBuffer();          // transfer internal memory to the display
      delay(5000);
      u8g2.clearBuffer();          // clear the internal memory
      u8g2.sendBuffer();          // transfer internal memory to the display
      delay(100);
    }
    else if (mySubString == "b")
    {
      u8g2.clearBuffer();          // clear the internal memory
      u8g2.setFont(u8g2_font_ncenB08_tr); // choose a suitable font
      u8g2.drawStr(0,10,"Amount added");  // write something to the internal memory
      u8g2.sendBuffer();          // transfer internal memory to the display
      delay(5000);
      u8g2.clearBuffer();          // clear the internal memory
      u8g2.sendBuffer();
      delay(100);
    }
  delay(100);
  }
  else {
    Serial.println(fbdo.errorReason());
  }
  delay(100);
  }
  rfid.PICC_HaltA();
  rfid.PCD_StopCrypto1();
  }
}
```

CHAPTER 5

Results and Conclusion

The implementation of the Smart Trolley system in supermarkets has yielded significant improvements in both customer experience and operational efficiency. By leveraging RFID technology and integrated sensors, the Smart Trolley has successfully automated the item tracking and billing process, providing real-time data for both customers and retailers.

Results:

- Reduction in Checkout Time: The elimination of manual scanning at checkout counters drastically reduced wait times, allowing customers to complete their purchases more quickly. This reduction in queues contributed to enhanced customer satisfaction.
- Accurate Billing: The integration of sensors for precise weight measurement ensured accurate billing, minimizing human errors that often occur during manual scanning.
- Real-time Inventory Updates: The Smart Trolley's ability to automatically update inventory levels as items are added or removed from trolleys enabled supermarkets to maintain accurate stock records. This feature facilitated improved inventory management, reducing instances of out-of-stock products and enhancing stock replenishment efficiency.

- Operational Efficiency: Supermarkets using the Smart Trolley system experienced an increase in operational efficiency, as employees were no longer burdened with manual scanning and repetitive tasks. This freed up resources, allowing staff to focus on other value-added tasks, such as customer service.

Conclusion:

The Smart Trolley system has successfully addressed key challenges in traditional shopping experiences, such as longer checkout times and inefficient inventory management. By automating item tracking and billing, the Smart Trolley has revolutionized the retail shopping process, providing faster, error-free transactions and real-time inventory updates. These innovations not only enhance the overall shopping experience for customers but also enable retailers to optimize their operations, improve stock management, and make informed decisions based on up-to-date inventory data.

In conclusion, the Smart Trolley offers a scalable solution for the retail industry, effectively transforming the shopping environment into a more convenient, efficient, and customer-centric experience.