

Pumped

Authors:

Mya Camacho-Hill | Computer Engineer

Simran Aulakh | Electrical Engineer

Alexandra Atuncar | Computer Engineer

Johan Gales | Electrical Engineer

Victoria Rodriguez | Sponsor

Matthew Aberman | Sponsor

Reviewer Committee:

Suboh Suboh | ECE

Dr. Nazanin Rahnavard | ECE

Mark Lundstrom | WUCF

University of Central Florida

Department of Electrical & Computer Engineering



LIST OF TABLES.....	VI
CHAPTER 1 EXECUTIVE SUMMARY.....	1
CHAPTER 2 PROJECT DESCRIPTION.....	2
2.1 PROJECT BACKGROUND AND MOTIVATION.....	2
2.1 EXISTING PRODUCTS.....	3
2.1.1 <i>Arduino Weighing Scale</i>	3
2.1.2 <i>Zebra Printers</i>	4
2.1.3 <i>Cubtale Smart Milk Inventory Tracking</i>	5
2.2 GOALS AND OBJECTIVES.....	6
2.3 DESCRIPTION OF FEATURES.....	7
2.4 ENGINEERING SPECIFICATIONS	8
2.5 HOUSE OF QUALITY.....	9
2.6 HARDWARE BLOCK DIAGRAM	9
2.7 SOFTWARE FLOWCHART.....	10
2.8 PROTOTYPE ILLUSTRATION.....	12
CHAPTER 3 RESEARCH AND INVESTIGATION	12
3.1 RELEVANT TECHNOLOGY.....	13
3.1.1 <i>Huckleberry</i>	13
3.1.2 <i>Baby Tracker</i>	13
3.1.3 <i>Nara Baby</i>	14
3.2 MICROCONTROLLER.....	14
3.2.1 <i>MSP430</i>	14
3.2.2 ARDUINO UNO R3.....	15
3.2.3 <i>RTK7RLG220C00000BJ</i>	16
3.2.4 <i>ESP32</i>	16
3.2.5 <i>STM32F411</i>	17
3.2.6 <i>Raspberry Pi Pico</i>	18
3.2.7 <i>Microcontroller Part Selection</i>	19
3.3 SCALE ARCHITECTURE.....	20
3.3.1 <i>Strain Gauge Load Cell Scales</i>	20
3.3.2 <i>Electromagnetic Force Restoration Scales</i>	21
3.3.3 <i>Capacitive Digital Scales</i>	21
3.3.4 <i>Hydraulic or Pneumatic Scales</i>	21
3.3.5 <i>Piezoelectric Digital Scales</i>	21
3.3.6 <i>Smart and IoT Connected Scales</i>	21
3.3.7 <i>Scale Part Selection</i>	22
3.4 WEIGHT SENSORS	22
3.4.1 <i>Load Cells</i>	22
3.4.1.1 <i>Single-Point Load Cells:</i>	22
3.4.1.2 <i>Bar-Type Load Cells</i>	23
3.4.1.3 <i>S-Type Load Cells</i>	23
3.4.1.4 <i>Compression Load Cells</i>	23
3.4.1.5 <i>Load Cell Part Selection</i>	25
3.4.2 <i>Load Cell Amplifier</i>	25
3.4.2.1 <i>HX711 Load Cell Amplifier</i>	25

3.4.2.2 ADS1232 Load Cell Amplifier	26
3.4.2.3 NAU7802 Load Cell Amplifier	26
3.4.2.4 Analog Load Cell Amplifier	26
3.4.2.5 Load Cell Amplifier Part Selection	28
3.5 PRINTER	29
3.5.1 Zebra ZQ620 / ZQ520.....	29
3.5.2 Brother RJ-2035B.....	29
3.5.3 Bixolon SPP-R310	30
3.5.4 MUNBYN ITPP941	30
3.5.5 Printer Part Selection.....	30
3.6 DISPLAY SCREEN	31
3.6.1 OLED (Organic Light Emitting Diode).....	31
3.6.2 LCD (Liquid Crystal Display).....	31
3.6.3 Electronic Paper Display	32
3.6.4 Multi-Segment LED Displays.....	33
3.6.5 LCD Part Selection.....	34
3.6.5.1 ILI9488 TFT Display	34
3.6.5.2 DT050BTFT	34
3.6.5.3 NHD-4.3-480272FT-CSXP-T.....	35
3.7 LEDs.....	36
3.7.1 Standard Through-Hole LEDs	36
3.7.2 Surface-Mount Device LEDs	36
3.7.3 Bi-color or Tri-color LEDs	37
3.7.4 WS2812B.....	37
3.7.5 LED Part Selection.....	38
3.8 POWER SUPPLY.....	38
3.8.1 Primary (Non-Rechargeable) Batteries.....	38
3.8.2 Alkaline Batteries	39
3.8.3 Zinc-Carbon Batteries.....	39
3.8.4 Silver Oxide Batteries	39
3.8.5 Secondary (Rechargeable) Batteries.....	39
3.8.6 Lithium-Ion Batteries.....	39
3.8.7 Lead-Acid Batteries.....	40
3.8.8 Nickle-Metal Hydride Batteries	40
3.8.9 Solar Power	40
3.8.10 Supercapacitors.....	41
3.8.11 Battery Part Selection.....	42
3.8.11.1 18650 Li-ion Battery	42
3.8.11.2 14500 Li-ion Battery	43
3.8.11.3 16340 Li-ion Battery	43
3.8.12 Part Selection Battery Charging IC/Module.....	44
3.9 VOLTAGE REGULATORS.....	45
3.9.1 LM2576	45
3.9.2 LM2679	46
3.9.3 LM2596	46
3.9.4 MP1584	46
3.9.5 XL4015.....	46
3.9.6 TPS561208DDCR.....	46
3.9.7 XL6009.....	47

3.9.8 MT3608	47
3.9.9 AMS1117.....	47
3.9.10 LM7805.....	48
3.9.11 LM3478.....	48
3.9.11 Voltage Regulator Selection	50
3.10 WIRELESS COMMUNICATION.....	50
3.10.1 Bluetooth	50
3.10.2 Wi-Fi	51
3.10.3 Zigbee	51
3.10.4 Ultra-Wideband	51
3.10.5 Near Field Communication (NFC).....	51
3.11 MICROCONTROLLER INTERFACES	51
3.11.1 PWM.....	51
3.11.2 ADC/DAC.....	52
3.11.3 Serial Communication Interfaces.....	52
3.11.4 UART (Universal Asynchronous Receiver Transmitter)	52
3.11.5 SPI (Serial Peripheral Interface)	53
3.11.6 I ² C (Inter-Integrated Circuit).....	53
3.11.7 Two Wire Automotive Interface	53
3.12 SOFTWARE RESEARCH	54
3.12.1 Mobile Operating Systems	55
3.12.2 Apple iOS.....	55
3.12.3 Google Android.....	55
3.12.4 HarmonyOS.....	56
3.12.5 KaiOS.....	56
3.13 PROGRAMMING LANGUAGES	56
3.13.1 C.....	56
3.13.2 Python (MicroPython).....	57
3.13.3 Arduino IDE	57
3.13.4 Visual Studio Code	57
3.14 MOBILE APPLICATION PROGRAMMING LANGUAGES	58
3.14.1 Java/Kotlin.....	58
3.14.2 Swift.....	58
3.14.3 React Native (JavaScript/Typescript)	59
3.14.4 Flutter (Dart)	59
3.15 CODE REPOSITORIES	60
3.15.1 GitHub	60
3.15.2 Bitbucket.....	60
3.16 DATABASES.....	61
3.16.1 SQLite	61
3.16.2 MySQL Database.....	61
3.16.3 Firebase Realtime Database.....	61
3.16.4 MongoDB	61
3.16.5 PostgreSQL.....	62
3.16.6 Database Selection	63
3.17 3D MODELING SOFTWARE	63
3.17.1 Tinkercad.....	63
3.17.2 Fusion 360 (Autodesk)	63

3.17.3 SolidWorks	64
3.17.4 FreeCad.....	64
3.17.5 OnShape.....	64
CHAPTER 4 STANDARDS AND DESIGN CONSTRAINTS	64
4.1 STANDARDS	64
4.1.1 Electrical and Power Standards.....	65
4.1.1.1 PCB Standards.....	65
4.1.2 Battery and Power Standards.....	67
4.1.3 LED Standards.....	68
4.1.3.1 IEC 62031	68
4.1.4 Wireless Communication Standards	68
4.1.5 Breastmilk Storage Standards.....	69
4.1.6 Security and Software Protocols.....	69
4.1.6.1 ISO/IEC 27001 (Information Security Management)	69
4.1.6.2 ISTQB Software Testing Standard.....	70
4.2 CONSTRAINTS.....	71
4.2.1 Testing Constraints.....	71
4.2.2 Time Constraints.....	73
4.2.3 Ethical and Social Constraints	75
4.2.4 Economic Constraints	76
4.2.5 Manufacturability Constraints.....	76
4.2.6 Realistic Design Constraints.....	77
CHAPTER 5 COMPARISON OF CHATGPT WITH OTHER SIMILAR PLATFORMS.....	77
5.1 PLATFORM COMPARISON.....	78
5.1.1 ChatGPT	78
5.1.2 Google Bard (Gemini).....	78
5.1.3 Microsoft CoPilot.....	79
5.2 BENEFITS	79
5.2.1 Specificity	79
5.2.2 Inspiration	79
5.2.3 Multimedia.....	79
5.3 DISADVANTAGES.....	80
5.3.1 Environmental Concerns	80
5.3.2 Inaccuracy	80
5.3.3 Academic Misuse	80
5.4 EXAMINING AI USAGE FOR OUR PROJECT	80
5.4.1 Case Study 1	80
5.4.2 Case Study 2	81
5.4.3 Case Study 3	82
CHAPTER 6 HARDWARE DESIGN.....	82
6.1 BREADBOARD DEMONSTRATION.....	82
6.2 OVERALL SCHEMATIC	83
6.3 SUBSYSTEM BLOCK DIAGRAM.....	84
6.4 CONVERTER SCHEMATICS.....	85
6.4.1 9V DC/DC Boost Converter Schematic.....	85

6.4.2 3.3V/5V DC/DC Buck Converter Schematic.....	86
6.5 MCU SCHEMATIC (USB TO UART BRIDGE)	89
6.6 TOUCHSCREEN SCHEMATIC	91
6.7 WEIGHT SENSOR SCHEMATICS	92
6.7.1 Load Cell Amplifier Schematic	93
6.7.2 Load Cell Schematic.....	94
6.8 LED SCHEMATIC	95
6.9 PRINTER	96
6.8.1 Printer to MCU Communication Module.....	97
CHAPTER 7 SOFTWARE DESIGN	98
7.1 MOBILE APPLICATION DEVELOPMENT	99
7.1.1 Mobile Application	100
7.2 DEVICE SOFTWARE	107
7.2.1 Load Cell Subsystem	109
7.2.2 LED Subsystem.....	109
7.2.3 Touchscreen Subsystem	110
7.2.4 Printer Subsystem.....	112
CHAPTER 8 SYSTEM FABRICATION/PROTOTYPE CONSTRUCTION.....	113
8.1 PCB LAYOUT	113
8.2 ENCLOSURE PROTOTYPE.....	115
CHAPTER 9 SYSTEM TESTING AND EVALUATION.....	119
9.1 HARDWARE TESTING.....	119
9.1.1 Load Cell Hardware Testing.....	119
9.1.2 Touchscreen Hardware Testing	121
9.1.3 Printer Hardware Testing.....	121
9.1.4 Power Supply Hardware Testing	122
9.2 SOFTWARE TESTING	122
9.2.1 Load Cell Software Testing	123
9.2.2 Touchscreen Software Testing	124
9.2.3 Printer Software Testing	124
9.2.4 Bluetooth Software Testing.....	125
9.3 PLAN FOR SENIOR DESIGN II	126
9.3.1 PCB Assembly.....	127
9.3.2 Software Development	127
9.3.3 Hardware Development.....	129
CHAPTER 10 ADMINISTRATIVE CONTENT	129
10.1 BUDGET	129
10.2 BILL OF MATERIALS	130
10.3 DISTRIBUTION OF WORK	132
10.4 MILESTONES	133
CHAPTER 11 CONCLUSION	135
APPENDIX A: REFERENCES.....	138
APPENDIX D: SOFTWARE CODE	142

APPENDIX E: CHATGPT PROMPTS & OUTCOMES.....	145
---	-----

List of Tables

Table 1: Prioritized Specifications	8
Table 2: Microcontroller Comparison.....	18
Table 3: ESP32 Microcontroller Comparison	20
Table 4: Load Cell Comparison	24
Table 5: Load Cell Amplifier Comparison.....	27
Table 6: HX711 Part Comparison.....	28
Table 7: Printer Comparison.....	30
Table 8: Display Screen Comparisons	33
Table 9: LCD Display Screen Comparison	35
Table 10: LEDs Comparison	37
Table 11: Power Usage Table	38
Table 12: Power Supply Comparison	41
Table 13: Lithium-Ion Battery Type Comparison	44
Table 14: Battery Charging IC/ Module Comparison	45
Table 15: Voltage Regulator Comparison.....	49
Table 16: Microcontroller Interface Comparison	53
Table 17: Programming Languages Comparison.....	59
Table 18: Database Comparisons.....	62
Table 19: Function and Variable Chart	106
Table 20: Estimated Pumped Project Budget.....	129
Table 21: Bill of Materials	130
Table 22: Distribution of Work.....	133
Table 23: Pumped Project Timeline	134

List of Figures

Figure 1: House of Quality.....	9
Figure 2: Hardware Block Diagram	10
Figure 3: Software Flowchart.....	11
Figure 4: Prototype Illustration	12
Figure 5: Overall Schematic	83
Figure 6: Subsystem Block Diagram.....	84
Figure 7: 9V DC/DC Boost Converter Schematic	86
Figure 8: DC/DC Buck Converter Schematic	87
Figure 9: Battery Charging Module Schematic.....	89
Figure 10: USB to UART MCU schematic.....	90
Figure 11: Touchscreen to MCU schematic	92
Figure 12: Load Cell Amplifier Schematic.....	94
Figure 13: Load Cell Schematic	95
Figure 14: LED to MCU schematic.....	96

Figure 15: MAX3232 to MCU schematic	98
Figure 16: First Time Access to App Flowchart	101
Figure 17: App Functionality Flowchart.....	101
Figure 18: App Prototype	102
Figure 19: Inventory Tracking System Flowchart	103
Figure 20: Settings System Flowchart	105
Figure 21: Preference Settings Flowchart	106
Figure 22: Use Case Diagram	107
Figure 23: Software Class Diagram	108
Figure 24: Homepage Draft	110
Figure 25: GUI and Load Cell Subsystem Integration	111
Figure 26: Touchscreen Flowchart	112
Figure 27: Main PCB.....	114
Figure 28: 3.3/ 5V Regulator PCB.....	114
Figure 29: Load Cell Amplifier PCB	114
Figure 30: LEDs/ Buttons PCB	115
Figure 31: Enclosure Sketch	117
Figure 32: Final CAD	118
Figure 33: Final Prototype.....	118

Chapter 1 Executive Summary

Breastfeeding mothers face many challenges including ensuring proper labeling of milk storage and having to keep track of milk inventory to avoid waste. The traditional method of manually labeling milk bags with a sharpie is both time-consuming and subject to errors, especially after late night pumping sessions. This Senior Design project aims to transform breastmilk management for mothers by developing a breastmilk scale and labeler that simplifies the tracking and storage process. Our device integrates an accurate digital scale, automatic thermal label printer, and an accompanying mobile application, offering an innovative solution for mothers, caregivers and healthcare professionals.

What sets our project apart from existing solutions is its integration of hardware and software, designed with user convenience in mind. The device accurately measure the volume of milk in the storage bags and print a label containing essential details such as date, time, volume, and optional dietary specifications. The device further enhances user convenience by syncing the data to the mobile application via Bluetooth after the printing of the label is completed. Furthermore, the mobile application increases usability by allowing users to manage milk inventory, expiration dates, and feeding patterns, reducing the risk of wasted milk.

Portability is also a key feature of our design, ensuring mothers can use the device both on-the-go and at home. A battery-powered structure allows for more flexibility in various environments while LED indicators provide real-time status updates, such as low battery alerts. Beyond individual benefits, our system also promotes milk donation by simplifying the labeling of dietary restrictions and allergens. This makes it easier for milk banks and hospitals to distribute donated milk to NICU babies and families in need. This device not only enhances convenience but also strengthens the support system for breastfeeding mothers.

This report documents the full design process, including system objectives, electrical integration, and both firmware and mobile application development. It follows the progression from prototype development to final implementation, incorporating rigorous testing to evaluate performance and refine the system based on results. By utilizing engineering knowledge with user-centered design, our project empowers mothers with an efficient, accurate, and stress-reducing solution for breastmilk management.

Chapter 2 Project Description

2.1 Project Background and Motivation

In the realm of innovative engineering, our senior design project seeks to address a challenge experienced by mothers everywhere. New mothers generally resort to breastfeeding as opposed to formula, as natural breast milk contains more nutrients and correlates to better immune system responses as the baby matures. Mothers aim to build a milk stash so that when they start work or travel, the baby can still receive the same amount of nutrients without reverting to formula. For pumped milk, it is important to label the date, time, and amount collected to ensure that it is consumed in a timely manner and does not expire or waste. While pumping milk for later use can save time and money, the process can be taxing. Currently the only way to label milk bags is with a sharpie and manually checking the number of ounces when pouring into the storage bag, as well as checking for the latest date and time. Although it may just seem tedious this process usually occurs late at night or early in the morning when the mother is tired. This can lead to errors when labeling, cause added stress, and be an inconvenience for the mother. Our project creates an efficient system for tracking and properly labeling breastmilk storage bags. This system combines practicality with ingenuity, empowering mothers with a solution that saves time, reduces stress and waste, as well as enhances their caregiving experience.

Our sponsors, Victoria Rodriguez and Matthew Aberman, approached our section and presented the idea of creating a device that would help with efficiently tracking breastmilk bags. After learning more about this idea, our group recognized the potential of this project to make a meaningful difference in the lives of mothers everywhere. Being able to save even a few minutes each day allows the mothers to feel some relief knowing that the milk has been accurately labeled, reducing guilt or anxiety when feeding their child. Victoria expressed that being able to keep track of how much milk their baby consumes is key when trying to avoid over and under feeding. She had shared that when many mothers are ready to go back to work after giving birth the process can be more complicated due to their breastmilk schedules overlapping with professional responsibilities. Having a portable device to take on the go and an application that keeps track of her milk supply provides a comforting feeling when a caretaker is with her child. Beyond alleviating a small portion of the stress for mothers, the product caters to mothers that are able to produce an excess amount of milk to consider donating to milk banks or hospitals. With the ability to easily label dietary restrictions and allergens on the milk it can easily be distributed for NICU babies, fosters, and families where the mother cannot produce sufficient amounts of milk.

This present difficulty not only requires technical skills but creativity and empathy to ensure the final product meets the needs of its users seamlessly. We believed strongly in this idea, knowing its immense value to future mothers, and took on this project to create something beneficial for women everywhere. Victoria and Matthew are both engineers which interested us as well knowing they can provide helpful feedback and new insights on the direction of our project.

Our project integrates software and hardware elements which challenged us to work cohesively together as a team. Guided by the feedback and expertise of our sponsors we were able to not

only create a solution that meets expectations but exceeds them. Through this venture we hope we've contributed to a brighter, more supportive future for mothers, families, and caretakers.

2.1 Existing Products

2.1.1 Arduino Weighing Scale

For our hardware design, our primary objective was to develop a portable digital scale that not only delivers accurate weight measurements but also seamlessly integrates with our software system for advanced functionality, including data tracking, labeling, and inventory management. In exploring different hardware implementation strategies, we came across a well-documented DIY Arduino-based weighing scale project on the maker platform Instructables. This project served as a foundational reference and provided us with valuable insights into the essential components required to bring our own concept to life.

The DIY project utilizes an Arduino Nano microcontroller and an HX711 load cell amplifier module, both of which are highly suited for compact and power-efficient designs, key factors that align with our goal of creating a portable solution. The Arduino Nano, in particular, is favored in this project for its lightweight form factor, low power consumption, and built-in USB interface for easy programming and debugging. These features make it an excellent candidate for embedded applications where space and power are limited, such as in handheld or mobile devices.

The HX711 module plays a critical role in the design, acting as an analog-to-digital converter (ADC) that allows the microcontroller to interpret the signals from a strain gauge-based load cell. Load cells produce small voltage changes in response to applied weight, and the HX711 amplifies and digitizes these signals with high precision. This module interfaces directly with the Arduino using a two-wire communication protocol (clock and data lines), simplifying integration while maintaining accuracy. This is especially beneficial in our design, where real-time weight measurements need to be both reliable and quickly processed for label printing and inventory tracking.

In the referenced build, the output from the HX711 is displayed on an LCD screen, providing users with immediate visual feedback on the measured weight. This is a design element we intend to mirror in our project; however, instead of a basic LCD, we've integrated a touchscreen display that can show not only weight but also additional data such as timestamps, milk bag information, and dietary notes. The touchscreen also serves as the user interface for interacting with system features like feeding suggestions and expiration tracking.

Beyond the basic scale functionality, we expanded upon the DIY project by incorporating wireless communication modules, such as Bluetooth or Wi-Fi, to sync weight data with our mobile or web-based application. This allows users to maintain an up-to-date inventory without manual logging.

Using the Instructables project as a reference gave us a strong starting point for prototyping. It not only clarified the electrical connections and circuit design involved in integrating a load cell

and amplifier with a microcontroller but also offered practical guidance on power supply options, calibration techniques, and sensor mounting. These insights helped us avoid common pitfalls and let us focus our efforts on refining and customizing the design to suit the specific needs of our application.

In summary, this Arduino-based weighing scale project was a crucial resource in laying the foundation of our hardware system. While we improved upon the design with additional features such as touchscreen UI and wireless data transmission, the core weighing mechanism remained similar. By building upon a proven framework and adapting it to our specific use case, we were able to deliver a compact, accurate, and user-friendly solution that bridges the gap between hardware functionality and advanced digital tracking.

2.1.2 Zebra Printers

Zebra printers are widely recognized for their versatility, reliability, and seamless integration across a variety of industries. Their robust capabilities make them an ideal choice for applications that require fast, consistent, and high-quality label printing—such as inventory management, healthcare, logistics, and retail. For our project, Zebra printers align particularly well with our needs, offering support for dynamic and detailed label content generation.

In our system design, we wanted to print labels that include critical information such as the date, time, weight of the milk, and any relevant dietary restrictions. Zebra printers are highly adaptable in this regard, as they are capable of printing both fully custom dynamic data and combining it with pre-printed templates, a feature that will allow us to maintain branding or standard formats while still personalizing each label for real-time use. This is especially beneficial in environments where clear, traceable, and standardized information is essential.

One of the standout advantages of Zebra printers is the company's extensive Software Development Kits (SDKs). These SDKs offer a wide range of tools that make integrating a Zebra printer into a custom hardware or software system more efficient and developer friendly. They provide well-documented Application Programming Interfaces (APIs), libraries, and sample code tailored to various platforms and programming languages. This means we can streamline the process of sending print commands directly from our microcontroller to the printer, reducing development complexity and ensuring greater system reliability. By leveraging Zebra's SDKs, we will be able to build a more robust and flexible software backend that handles label formatting, printer status monitoring, and connectivity management.

Additionally, Zebra Technologies has already made inroads in the field of breast milk tracking, offering barcode labels specifically designed for this purpose. These labels typically include crucial details such as the date and time of expression, volume, and expiration date—information that is essential for caregivers to manage breast milk safely and effectively. Such labeling systems not only ensure that the milk is used in a timely manner but also enhance safety by reducing the risk of expired milk being fed to an infant. The simplicity of barcode scanning makes these labels extremely accessible, allowing even non-technical caregivers to interact with and interpret the information with ease.

The use of barcode labeling is a particularly attractive feature for our own design. By implementing barcode functionality, we enable users to quickly scan a milk bag and retrieve or

log information within our system, reducing manual input and minimizing the risk of human error. This could also enhance the overall accessibility of our product, allowing parents, caregivers, or healthcare providers to interact with the system in an intuitive and user-friendly way.

Furthermore, integrating Zebra printers aligns with our broader project goals of creating a professional, scalable, and future-proof solution. Zebra's reputation for hardware durability and long-term support adds value to our design, especially if the product were to be commercialized or expanded to support multiple use cases in the future.

In summary, Zebra printers provide a well-rounded solution that supports our technical requirements and end-user experience goals. Their ability to handle dynamic data printing, their robust SDK offerings for seamless software integration, and their preexisting involvement in breast milk tracking all make them an ideal fit for our project. By incorporating Zebra's technology into our system, we can ensure high-quality, consistent, and accessible label printing, making our device more efficient, trustworthy, and user centric.

2.1.3 Cubtale Smart Milk Inventory Tracking

Cubtale is a company that offers innovative and thoughtful solutions to everyday challenges faced by mothers, particularly around infant care and feeding. One of their standout products is the "Smart Milk Inventory Tracking – Labels," which blends physical labeling with digital tracking to provide a streamlined solution for managing breastmilk inventory. These smart labels include a barcode along with manually recorded details such as the date, time, volume, name, and milk bag number. Once a label is applied, users can input this data into Cubtale's accompanying mobile app by referencing the unique label number.

The app enhances the user experience with a suite of intelligent features. For instance, it offers feed reminders that suggest which milk bag to use based on expiration priority, helping to minimize waste. It also notifies users when a milk bag is approaching its expiration date, ensuring that milk is used while it's still safe and effective. Additionally, the app provides educational resources such as breastmilk storage guidelines, which empower parents and caregivers with best practices for handling, storing, and using breastmilk. Users can monitor their milk inventory in real-time, allowing them to track supply levels and avoid running out unexpectedly.

Beyond milk tracking, Cubtale's product line includes Cubtale Cubs, compact smart devices that monitor and log daily baby care activities. These devices are purpose-built for simplicity: each Cub has a single button that corresponds to a specific task, such as tracking sleep, diaper changes, medicine, personal care, or pump use. When pressed, the button logs the activity and sends the data to the app. In scenarios where Wi-Fi is unavailable, the Cubs can store the last five recorded activities in their internal memory and sync them once the connection is re-established. This offline data retention capability is particularly inspiring and directly relevant to our own project; we would like to implement a similar functionality in our scale to ensure data integrity even during temporary Bluetooth disconnections.

Another compelling feature of Cubtale Cubs is their brief connectivity period to Wi-Fi, which significantly reduces radiation exposure, an important consideration when designing products for infants. This thoughtful design choice prioritizes child safety, and we are exploring similar methods to limit wireless exposure by designing our device to connect to Bluetooth or Wi-Fi only during brief, necessary data transfers. Doing so will help conserve power, reduce background radiation, and maintain user trust.

Drawing inspiration from Cubtale's ecosystem, we aim to incorporate several of their key functionalities into our own project. Specifically, we plan to develop a software interface that not only logs milk bag data but also provides expiration date alerts and suggests which milk bag to use next, based on a first-in-first-out approach. Barcode-based labeling could be central to our system design, enabling users to scan a milk bag for quick and accurate logging. This integration will streamline data entry, reduce the chance of manual error, and enhance the user experience.

Moreover, our system will be designed with multiple caregivers in mind. By allowing real-time updates and activity logs, similar to Cubtale's notification system, we will provide parents peace of mind, knowing their child's care routine is being followed accurately. These features will make our project not just a scale, but a comprehensive, user-friendly tool for managing infant feeding with precision and convenience.

In summary, Cubtale offers an excellent model of how to combine hardware simplicity with intelligent software, and their thoughtful design decisions serve as a strong reference for the development of our own product. By adapting and building upon these principles, we hope to create a device that meets the real-world needs of parents and caregivers with the same level of reliability, safety, and user-centered design.

2.2 Goals and Objectives

Our goals are organized into three tiers: basic, advanced, and stretch. The basic goals focus on fundamental functions to assist mothers in breastmilk management. The advanced goals expand on this foundation by adding more interactive features to the device. Stretch goals further enhance the system by integrating data-driven insights and multi-user accessibility.

Goals:

Basic Goals:

- To help mothers manage pumped breastmilk
- Design a device that measures pumped breastmilk
- Have the device print a label for breastmilk
- Creating a user-friendly interface for mothers to access information

Advanced Goals:

- Optimize design for portability
- Offer customizable labeling
- Help monitor milk inventory
- Design a physical guide system to minimize milk spillage.
- Provide visual feedback on system status

Stretch Goals:

- Provides insights to help mothers make informed dietary decisions.
- Implement multi-user access with varying levels of permission.
- Enhance software for integration with milk banks.

Objectives:

To enable accurate tracking and management of breast milk storage, the device measures and logs milk volume after each pumping session. A built-in sensor determines the exact volume of milk with consideration for the weight that may come from the packaging used for the milk. Labels include date, time, & weight received from the scale, with optional input for allergens or dietary restrictions. Using a pre-built thermal printer, a label is printed with the data collected from the scale. Advanced labeling options allow users to enter additional details such as donation specific fields for more personalization. The system utilizes labels that adhere to milk bank and hospital requirements. The user-interface then allow users to manually edit details in the mobile application, if needed.

To prioritize portability, the device was designed with compact and lightweight hardware components, which will enable mothers to carry it while traveling or pumping on the go. In addition, the shape of the scale is ergonomic, allowing it to support the bag of breastmilk to minimize spillage. A battery-operated structure ensures this convenience without sacrificing its functionality, and the implementation of LEDs provide visual feedback for the connection status of different peripherals.

The software that accompanies this device manages inventory, tracks expiration dates, and notifies users of inventory issues; ensuring the efficient and safe use of breastmilk. These features are provided through a user-friendly app, allowing mothers and caregivers to make informed decisions regarding the child's intake. Additionally, the app analyzes patterns from the data entered such as feeding frequency, baby's age, and amount stored in inventory to help mothers anticipate future milk needs. By integrating accurate hardware measurements with software capabilities, this system provides a solution that meets the needs of breastfeeding mothers and their support networks.

2.3 Description of Features

Our sponsor has provided us with a list of primary user jobs and requirements for this project. In our design, we are aiming to solve the problems presented to us for breastfeeding moms as well as caregivers and healthcare professionals.

Breastfeeding moms must not only mentally track but label all of the milk they produce for their child. This can become tricky, as sometimes mothers lose track of when they have produced the milk and how much of it has been produced as well. With the scale feature, we have fixed this problem. In our design we created a scale that can weigh the volume of the milk, and then print the date, time, and weight on a label. Sometimes mothers must make sure they label the bags based on allergy specifications, such as writing if it is dairy-free. In our design, we added a feature that prints these specifications on the same label.

In collaboration with the device, we developed an app that makes keeping track of milk inventory seamless. Utilizing the device after entering the proper information and printing the label, the device then logs the new quantity of milk into the inventory stored in the app. Through this app users can edit the details of the milk inventory as well as delete from it as the milk is being used. Other features of this app include the ability to receive notifications when milk is about to expire and check-ins for if any milk was properly logged out of inventory. This application is be accessible at the mother's discretion where she can create secondary accounts for caregivers that will allow them access to her data but provide restrictions for that user's access too. Since this app was not only designed for mothers as users it also has a feature that calculates the necessary amount of milk needed for feeding based on user input. Utilizing this information, it is also able to determine when more milk is necessary to meet feeding needs. The user can set a threshold of when they want to be notified of when inventory is getting low. Every baby's feeding schedule is different so this feature helps account for that.

2.4 Engineering Specifications

At the beginning of our project, we made informed estimates on our values based on research of previous projects/products. For our testable specifications, we decided on testing weighing accuracy, LEDs, and display.

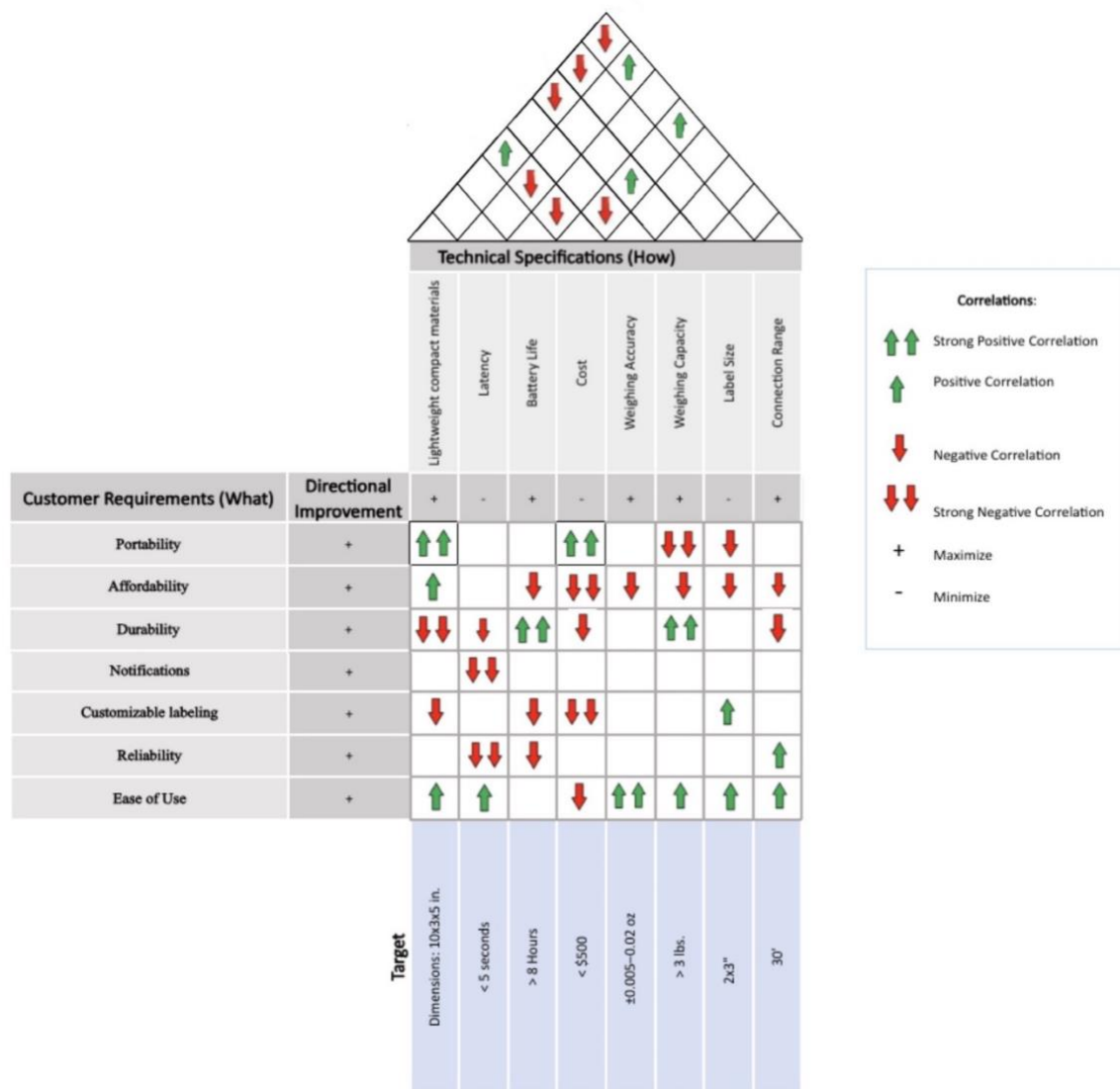
Table 1: Prioritized Specifications

Parameter	Description	Value
Weight	Weight of product	1-5 lbs.
Dimension	Total volume of product must adhere to portability requirements	10 x 5 x 3 inches
Storage Capacity	Total logs of milk bags that can be stored in data inventory	1200 Entries
Battery Life	Rechargeable battery must charge scale and printer	> 9 Hours
Weighing Accuracy	Feedback from scale must be accurate	± 0.2 oz
Maximum Weight Capacity	Max amount of weight the scale can hold, must account for breastmilk bags of larger volumes	1-11 lbs.
Dimension of Labels	FDA compliant, has date/time/dietary restrictions	1.2 x 2-inch labels
Wireless Connection Range	Range of Bluetooth connection between product and mobile device	30 feet

Software Response Time	Response time to see data from hardware device to mobile application	< 5 seconds
Printing Time	Time it takes to print one label	< 8 seconds

2.5 House of Quality

Figure 1: House of Quality

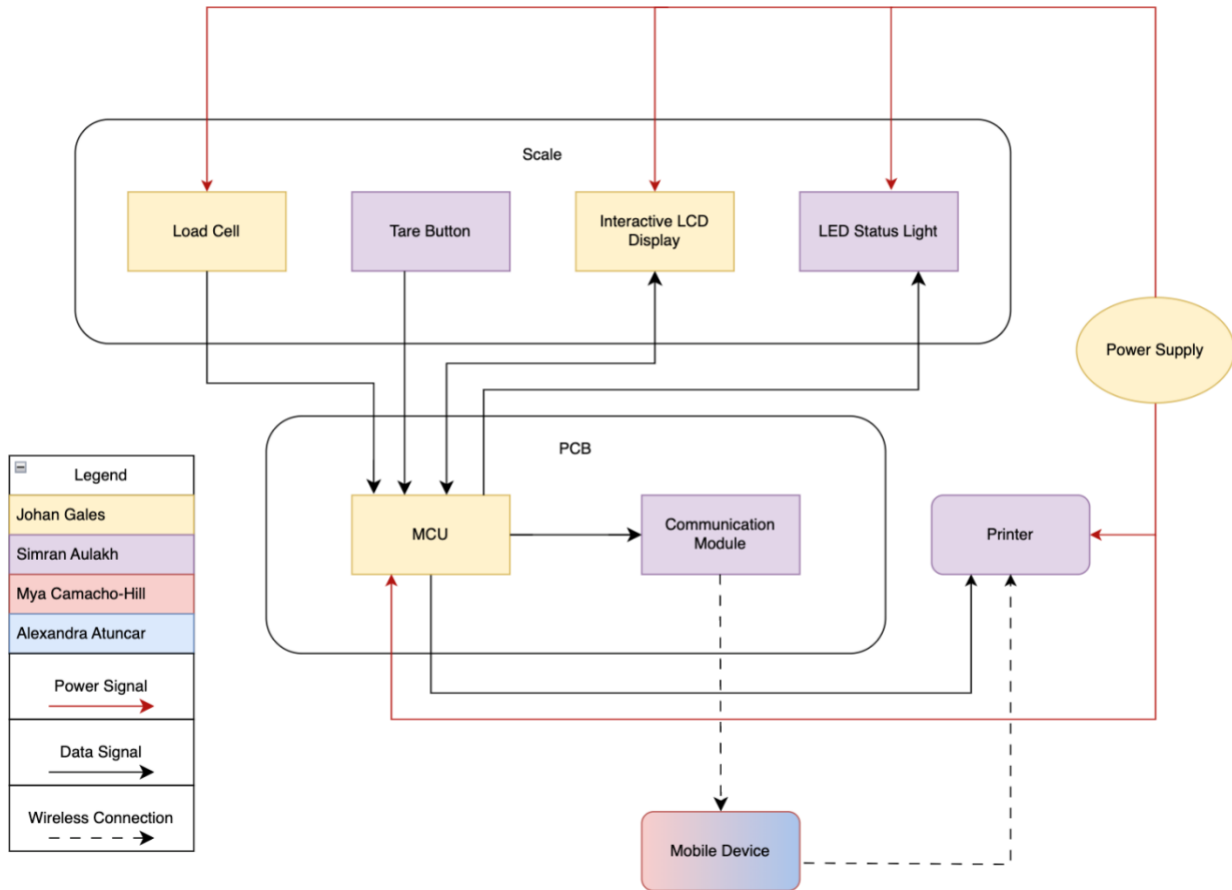


2.6 Hardware Block Diagram

For our prototype, we implemented all circuitry on a custom-designed printed circuit board (PCB) to streamline the integration and reliability of our device. The PCB houses key components including the microcontroller (MCU), load cells for weight sensing, a tare button,

and the communication interface that connects to the label printer. The hardware block diagram shown here provides a high-level overview of how major subsystems interact. Each component is visually grouped to reflect its role in the overall system. This block diagram helps clarify the flow of power and data signals throughout the device, making it easier to conceptualize, troubleshoot, and refine each subsystem independently before moving on to physical PCB design.

Figure 2: Hardware Block Diagram

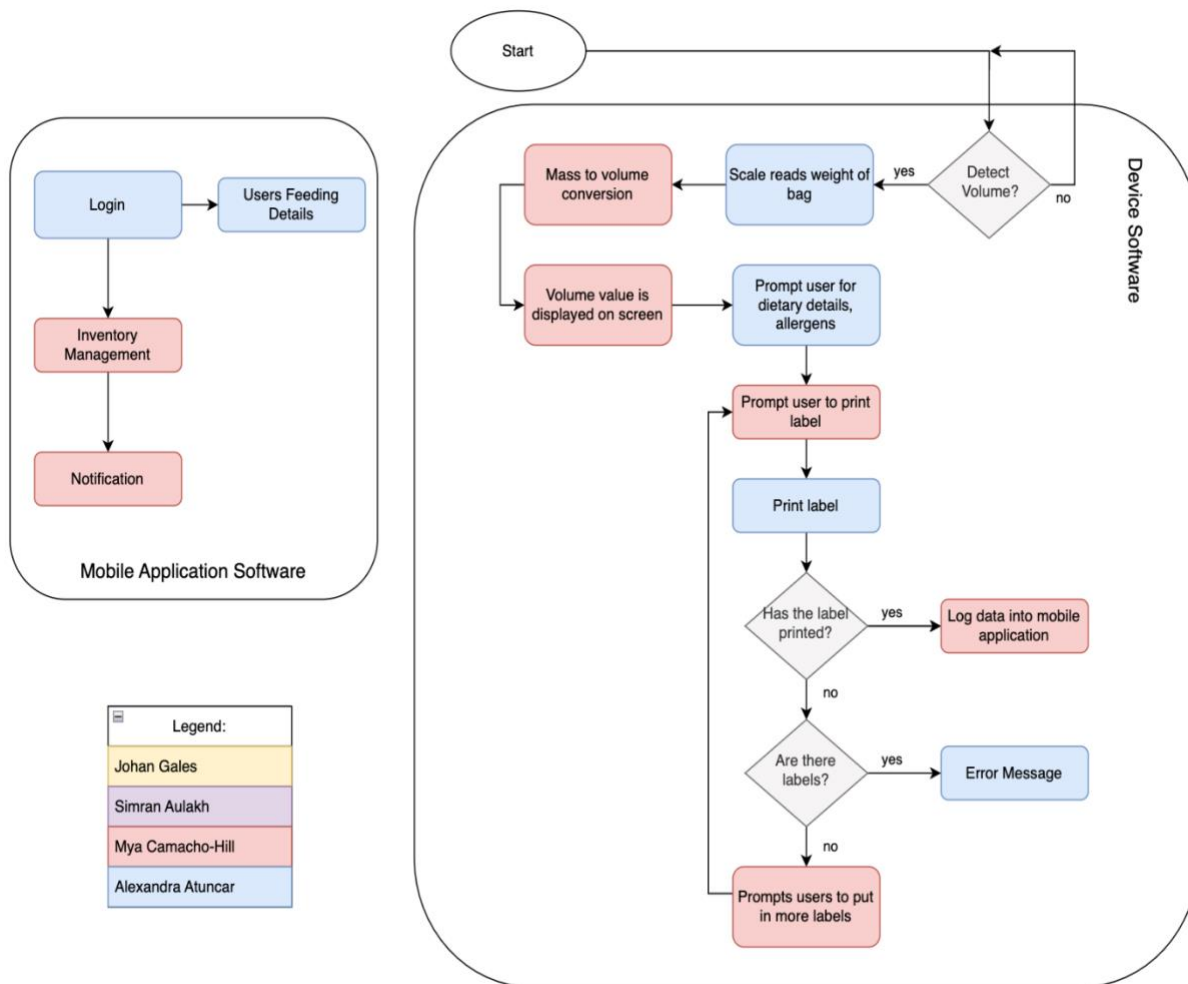


2.7 Software Flowchart

Our software flowchart illustrates the process by which the device captures, labels, and logs breastmilk data. The device's software begins by detecting the presence of a milk bag and reading its weight. It then converts this weight into volume, prompting the user for any additional dietary information or allergen notes, and generates a label for printing. Once the label is successfully printed, the data is logged into the mobile application. The mobile app enables users to manage their milk inventory, track storage and expiration, and enter child specific feeding preferences. Additionally, it provides notifications to remind users of important actions,

such as low inventory, or expired milk. This integrated system supports both the accurate tracking of breastmilk and the personalization of care for each child's dietary needs.

Figure 3: Software Flowchart



2.8 Prototype Illustration

Figure 4: Prototype Illustration



Chapter 3 Research and Investigation

In this section, we will explore and evaluate various hardware components and software tools that can be integrated into our system. This research allows us to compare different options based on key factors such as performance, compatibility, power efficiency, cost, and ease of implementation. By examining the strengths and limitations of each component and software platform, we aim to make informed decisions that align with the functional and technical requirements of our project.

Our evaluation will cover system elements such as microcontrollers, power regulators, wireless communication modules, user interface components, and thermal printers, as well as development environments, programming languages, and mobile app frameworks. Each choice will be assessed not only in isolation but also in terms of how well it integrates with the overall system architecture.

The outcome of this research will guide us in selecting the most suitable technologies to ensure the reliability, efficiency, and scalability of our design. Ultimately, this process supports the development of a cohesive, high-performing solution tailored to the specific goals of our wireless, portable digital scale with integrated label printing.

3.1 Relevant Technology

Currently there are a number of apps that act as data trackers for mothers to use in various aspects of their lives. In relation to our project specifically, some of these apps include inventories that can be used to keep tabs on available stored breastmilk as well as pumping times.

3.1.1 Huckleberry

Huckleberry is an all-around tracker app for various aspects relating to babies/mothers including sleep, feeding, bathroom use, pumping, medicine, growth, and activities. Under each of these sections, mothers can record the times of the day that the baby has done the category (slept, peed, ate, etc.), with additional prompts for the quality of these actions, other descriptors, and personal notes. For every time-based activity, the data recorded is then displayed in a weekly schedule with blocks for the duration of each activity. There are also tabs for activity overviews by day, in a list, or as a summary.

Relevant to our project is the pumping category that allows users to log how much milk is being pumped (able to be distinguished further between left and right breast or as a total), by ounce or by milliliter, the time it began being pumped, and the duration of the pump session. This data is then recorded on the same schedule as previously mentioned, however it cannot access or track specific bags of the milk being pumped, only the time and amount that was pumped. While these features are undoubtedly useful to mothers, the more important aspect for our project is keeping track of how much milk is available to the mother, as well as when it can be used by and stored. The Huckleberry program is also only available through a monthly or annual subscription that begins with a 14-day free trial.

3.1.2 Baby Tracker

Baby Tracker holds many similarities to Huckleberry in its ability to track and monitor several aspects of parenting, including feeding (nursing, formula, supplement), diaper use, sleep schedule, physical growth, milestones, and more. As it relates to our project, the app allows mothers to track their pumped breastmilk: the feature includes a toggle that switches between live timed logging or retroactive logging, as well as the ability to distinguish between left and right breast pumping. The collected milk is logged in ounces, and each log automatically updates the overall amount in the inventory. In another tab, the activity logs can be viewed by day, with options to make a note about each activity (or pump) and include facts like allergens or other warnings. The app is completely free and utilizes a very minimalist style that is easy to follow and understand.

This app serves as a valuable reference for our project in terms of both functionality and user experience. Its ability to track milk volume and organize data into intuitive daily logs provides a strong example of how we might structure our own data tracking and display system for pumped milk. The toggle between live and retroactive logging is particularly relevant, as it mirrors potential use cases for our device which allows users to either record the milk weight in real time or manually input it later. Additionally, the simplicity and clarity of the interface offers insight into how we might design our touchscreen UI to prioritize ease of use, especially for sleep-deprived or time-constrained caregivers.

3.1.3 Nara Baby

Nara Baby holds many of the same capabilities of the previously mentioned tracker apps with a few notable differences: firstly, the specific fields of activity tracking can be toggled on or off, curating the app experience to better suit the parent's needs. This allowed us to fixate purely on the breastmilk aspect, ignoring all the other activities irrelevant to our project. The second notable difference from the previous two apps was the ability to add caretakers and join existing family accounts, a feature that is highly relevant to our goals with the project.

The customizable activity tracking in Nara Baby offers a clear model for how our device interface could be tailored to individual user preferences, allowing caregivers to focus solely on milk-related data without being overwhelmed by unrelated functions. This aligns with our goal of creating a streamlined, purpose-built tool for milk tracking. Moreover, the ability to sync data across multiple users through family accounts presents an inspiring framework for future scalability, which is to enable shared access between partners, caregivers, or healthcare providers. This collaborative feature supports our vision of a connected system where breastmilk inventory and usage logs can be updated and accessed by multiple trusted individuals, enhancing convenience and ensuring continuity of care.

While these apps have an overlap with the software aspect of our project, we aim to further integrate the physical labelling and weighing of the breastmilk being collected through our hardware, creating a device that will further facilitate the pumping process for mothers in conjunction with an app for inventory. At the very least, these apps can help inform the structure of our app and software.

3.2 Microcontroller

To facilitate communication between the various subsystems in our design, we are utilizing a microcontroller as the central control and communication hub. The microcontroller plays a pivotal role in managing data flow between the load cell amplifier, touchscreen display, and label printer, ensuring synchronized operation across the system.

One of the key functions of the microcontroller is to enable wireless communication with our mobile application via built-in Wi-Fi and Bluetooth connectivity. In addition to real-time communication, the microcontroller is also responsible for logging weight readings and transmitting them to the mobile application or an external server for record-keeping, analytics, or inventory management purposes.

3.2.1 MSP430

The MSP430 is a 16-bit microcontroller family developed by Texas Instruments, known for its ultra-low power consumption and efficient performance in battery-operated systems. It is specifically designed for applications requiring extended battery life. The MSP430 features a selection of advanced low power operating modes, allowing the system to conserve energy during idle times and wake up quickly to perform tasks, thereby significantly extending battery life.

In terms of peripheral integration, the MSP430 is well-equipped with a comprehensive set of on-chip modules, including timers, UART, SPI, I²C, watchdog timers, and multiple analog-to-digital converters (ADC). These built-in peripherals make it particularly effective for interfacing with sensors, such as load cells in a digital scale application. Its ADC modules provide high-resolution analog signal acquisition, which is essential for accurate and reliable weight measurements.

Despite its advantages in power efficiency and sensor interfacing, the MSP430 presents certain limitations. Its processing power is not as high when compared to modern 32-bit microcontrollers like the ESP32. This limits its ability to handle graphical user interfaces, such as touchscreens, or to manage resource-intensive tasks like real-time data processing, USB host control, or controlling a label printer, especially if complex protocols are involved. Another challenge is the development environment and learning curve. While Texas Instruments provides development tools such as Code Composer Studio and MSP430Ware, these tools may be less accessible to beginners compared to the Arduino or ESP32 ecosystems. Additionally, the smaller community and less widespread third-party support may result in a more challenging development curve, particularly when implementing peripherals outside the basic use case.

In conclusion, the MSP430 is an excellent choice for power-sensitive applications where efficiency and sensor data acquisition are the primary focus. However, due to its limited processing capabilities, lack of built-in wireless communication, and increased development complexity, it may not be ideal for more sophisticated embedded systems that require rich user interfaces or advanced peripheral control, such as the touchscreen-driven, wireless-enabled label printing scale envisioned for this project.

3.2.2 Arduino UNO R3

The Arduino UNO R3 is an 8-bit microcontroller development board based on the ATmega328P. One of its key strengths is its user-friendly Integrated Development Environment (IDE), which simplifies code development, debugging, and uploading, making it particularly useful for beginners.

The UNO R3 features 14 digital I/O pins, 6 analog inputs, and communication interfaces such as UART, SPI, and I²C, enabling basic connectivity with sensors and peripheral devices. It is often favored in early development because of the abundance of libraries and documentation, which help accelerate testing and prototyping.

However, its 8-bit architecture and limited resources significantly constrain its performance for more demanding applications. With only 2 KB of SRAM, 32 KB of Flash memory, and a 16 MHz clock speed, the UNO R3 struggles to handle complex tasks such as real-time data processing, touchscreen management, or multitasking. Furthermore, it lacks built-in wireless communication, requiring additional external modules for connectivity, which increases both complexity and power consumption.

In the context of this project, which involves real-time weight measurements, a touchscreen user interface, and wireless data transmission, the Arduino UNO R3 falls short in both processing capacity and feature integration. While it remains a tool we are considering for basic prototyping

or sensor testing, it is not a practical choice for the final product, especially one requiring high performance, interactivity, and modern communication capabilities.

In summary, the Arduino UNO R3 offers a great entry point for learning and early development, but due to its hardware limitations and lack of integrated wireless features, it is not well-suited for complex, performance-intensive embedded systems such as the final design of this project.

3.2.3 RTK7RLG220C00000BJ

The RTK7RLG220C00000BJ, based on the Renesas RL78/G22 microcontroller, is a 16-bit development kit optimized for ultra-low power applications, making it particularly well-suited for battery-powered and energy-sensitive devices. With its industry-leading low-power operation, the RL78/G22 excels in scenarios where maximizing battery life is critical.

A key feature of this development kit is its integrated capacitive touch sensing unit, which enables intuitive and responsive user interfaces without the need for additional touch controller ICs. This built-in functionality not only reduces hardware complexity but also helps conserve board space and cost. In addition, the RL78/G22 supports multiple serial communication protocols, including I²C, SPI, and UART, offering sufficient flexibility for connecting a variety of sensors and peripherals.

The 16-bit CPU architecture has a balance between performance and power efficiency, providing reliable processing capability for tasks such as sensor data acquisition and basic user interface management, while keeping energy consumption minimal. It also offers on-chip safety features and multiple timers, which enhance its reliability for embedded control applications. However, the RL78/G22 comes with certain limitations. Its lack of built-in wireless communication necessitates the use of external modules for any form of wireless data transmission. This not only adds to the hardware cost and power budget but also increases software complexity due to the need for additional communication drivers and protocols. Furthermore, integrating and configuring peripherals, especially more complex ones like touchscreen displays or wireless modules, can be more time-consuming and require additional development effort compared to higher-level platforms like the ESP32.

In summary, the Renesas RL78/G22 Kit offers exceptional energy efficiency and solid peripheral support for basic embedded applications, making it a strong candidate for designs where power consumption is a primary constraint. However, for applications involving wireless communication or complex UIs, it may require more extensive development and external components.

3.2.4 ESP32

The ESP32 is a powerful 32-bit microcontroller that integrates Wi-Fi and Bluetooth capabilities, making it an excellent choice for applications that require wireless communication, such as remote data transmission or cloud connectivity. Its dual-core processor, running at clock speeds up to 240 MHz, provides ample computational power to efficiently manage multiple concurrent tasks, including real-time weight measurements, touchscreen user interface management, and data logging or wireless transfer.

One of the ESP32's standout features is its versatile peripheral support. It offers a wide array of GPIO pins, along with multiple interfaces including I²C, SPI, UART, PWM, and ADC, which allows seamless integration with various external components like load cells, touchscreen displays, and label printers. This flexibility significantly simplifies the hardware design and allows for future upgrades or additional features.

However, it is important to note that the ESP32's power consumption is higher compared to ultra-low-power microcontrollers such as the MSP430 or RL78/G22, which are specifically designed for energy-efficient, battery-operated applications. In scenarios where battery life is critical, careful power management, such as utilizing the ESP32's deep sleep modes, would be necessary to optimize energy usage.

Despite this, the ESP32's superior processing capabilities, rich connectivity options, and broad community support make it a well-rounded and future-proof platform for this project. It strikes a strong balance between performance and flexibility, making it particularly suitable for complex embedded systems like a portable digital scale with wireless communication and a touchscreen interface.

3.2.5 STM32F411

The STM32F411, part of the STM32F4 series from STMicroelectronics, is a 32-bit ARM Cortex-M4 microcontroller known for its high-performance processing and energy efficiency. It offers a balance between computational capability and power consumption.

Operating at clock speeds up to 100 MHz, the STM32F411 delivers significantly more processing power than 8-bit or 16-bit microcontrollers, making it suitable for handling complex tasks such as touchscreen interfaces, real-time sensor data processing, and controlling peripherals like label printers. It also includes a floating-point unit (FPU), which is particularly useful for precise and efficient mathematical computations, valuable for applications such as weight calibration and filtering of sensor data.

The STM32F411 boasts a wide range of integrated peripherals, including I²C, SPI, UART, USART, PWM, ADC, and DMA (Direct Memory Access), allowing seamless integration with sensors, actuators, and user interfaces. Its large memory resources offer support for more demanding applications, such as running a touchscreen GUI library or buffering data for printing or transmission.

One of the key trade-offs is the lack of built-in wireless connectivity. Unlike the ESP32, the STM32F411 does not include Wi-Fi or Bluetooth, so external communication modules must be added if wireless features are required. While this introduces some additional hardware and software complexity, it can result in lower overall power consumption, since developers can select and control the wireless module independently, tailoring its usage to the application's specific power constraints. Moreover, while the STM32 is powerful and widely used in industrial and professional applications, it has a steeper learning curve than platforms like Arduino.

In summary, the STM32F411 is a strong candidate for embedded applications that demand a balance of performance and power efficiency. Its ability to handle touchscreen interfaces, sensor inputs, and complex peripheral control, combined with the option to add selective wireless connectivity, makes it a flexible and scalable solution for this project.

3.2.6 Raspberry Pi Pico

The Raspberry Pi Pico is a versatile and cost-effective microcontroller board based on the RP2040 chip. It delivers impressive computational performance relative to its size and price point, making it well-suited for a wide range of embedded applications. The dual-core architecture allows for efficient multitasking, enabling it to simultaneously manage sensor inputs, user interface elements, and control logic.

With 264 KB of SRAM and 2 MB of onboard Flash memory, it provides ample space for storing firmware, graphical interface assets, and buffer data. Its compatibility with a variety of display libraries and interfaces, such as SPI and I²C, makes it easy to connect and control touchscreens, creating a smooth and responsive user interface for intuitive operation. Additionally, the board supports multiple GPIOs, PWM channels, ADC inputs, and serial communication interfaces, offering the flexibility needed to connect and manage load cells, amplifiers, and other peripherals.

However, one notable limitation of the Raspberry Pi Pico is its lack of built-in wireless connectivity. To enable wireless communication, external modules must be added. While this increases hardware and software complexity, it also gives developers the freedom to choose modules that best suit their power and performance needs. This can be beneficial in applications where wireless communication is occasional.

In conclusion, the Raspberry Pi Pico is a powerful, flexible, and budget-friendly microcontroller that offers strong performance for applications requiring real-time processing, display integration, and sensor handling. Although it requires external components for wireless communication, it remains an option for this project, especially when the design prioritizes performance and display capabilities over built-in connectivity.

Table 2: Microcontroller Comparison

Microcontroller	MSP430(TI)	Arduino UNO R3	RTK7RLG2 20C00000B J (Renesas RL78/G22)	ESP32	STM32F411	Raspberry Pi Pico
Architecture	16-bit	8-bit	16-bit	32-bit	32-bit	32-bit
Clock Speed	Up to 16 MHz	16 MHz	32 MHz	Up to 240 MHz	100 MHz	133 MHz

RAM	2-8 KB	2 KB	4-8 KB	520 KB	128 KB	264 KB
Flash Memory	16-128 KB	32 KB	64 KB	4 MB	512 KB	2 MB
Connectivity	None	None	None	Wi-Fi, Bluetooth	None	None
I/O Interfaces	UART, SPI, I2C, ADC	UART, SPI, I2C, ADC	UART, SPI, I2C, ADC	UART, SPI, I2C, ADC, DAC, PWM	UART, SPI, I2C, ADC, DAC, PWM	UART, SPI, I2C, ADC, PWM
Power Consumption	Ultra-low power	Moderate	Ultra-low	Higher power consumption	Low Power	Low Power
Operating Voltage	1.8-3.6V	5V	1.6-5.5V	2.2-3.6V	1.7-3.6V	1.8-3.3V
Temperature Range	-40-85°C	0-70°C	-40-85°C	-40-125°C	-40-105°C	-20-85°C

3.2.7 Microcontroller Part Selection

The microcontroller selected for our project is the ESP32, specifically the ESP32-WROOM-32, chosen for its robust features and versatility. One of the primary reasons for choosing the ESP32 is its built-in wireless and Bluetooth connectivity, which is crucial for our project. This capability allows the milk data logged on the device to be wirelessly transferred to the app via Bluetooth. Additionally, the ESP32 supports a wide range of communication interfaces, including SPI and I2C, which is beneficial given the variety of hardware components we've integrated into the system. The flexibility to use multiple communication protocols ensures that we can easily interface with different components as needed. Finally, the ESP32 operates at a voltage level compatible with our chosen power supply, making it a perfect fit for the overall design.

Table 3: ESP32 Microcontroller Comparison

Feature	ESP32-WROOM-32	ESP32-WROVER	ESP32-S3	ESP32-C3
Antenna Type	Onboard PCB	PCB Trace/ External	PCB Trace/ External	PCB Trace/ External
CPU	Dual-core Xtensa LX6	Dual-core Xtensa LX6	Dual-core Xtensa L7	Single-core RISC-V
Clock Speed	Up to 240 MHz	Up to 240 MHz	Up to 240 MHz	Up to 160 MHz
RAM	520 KB SRAM	520 KB SRAM and 4MB PSRAM	512 KB and optional PSRAM	400 KB
Flash	4 MB/ 16 MB	4 MB/ 16 MB	Up to 8 MB	4 MB (typical)
Wi-Fi	802.11 b/g/n	802.11 b/g/n	802.11 b/g/n	802.11 b/g/n
Bluetooth	Bluetooth v4.2 (BR/EDR and BLE)	Bluetooth v4.2	Bluetooth 5 LE	Bluetooth 5 LE
USB Interface	No	No	Yes (native USB)	Yes (via USB-Serial)
GPIO Pins	Up to 34	Up to 34	Up to 45	Up to 22
Power Consumption	Moderate	High	Low to Moderate	Low
Best For	Long-range communication	High-memory apps	AI and USB apps	Compact low-power designs

3.3 Scale Architecture

To support our core functionality of measuring breastmilk volume by weight, we are integrating a precision scale feature directly into our design. Accurate weight measurement is essential in this application, as even small variations in weight can provide valuable insight into an infant's nutritional intake. Selecting the appropriate scale architecture is therefore critical to achieving both accuracy and reliability.

3.3.1 Strain Gauge Load Cell Scales

This type of scale uses strain gauge load cells, which are sensors that measure strain in response to the applied weight. When weight is applied, the metal structure deforms, causing a change in the electrical resistance of the strain gauge. This change in resistance is converted into an electrical signal, which is then processed by an Analog-to-Digital Converter and displayed as weight. Strain gauge load cell scales are commonly used in retail settings, warehouses, factories, and for personal weighing. Some advantages of this scale include its wide availability, reliable accuracy, and ability to handle a broad range of weights. However, some drawbacks include its sensitivity to temperature fluctuations and the potential for damage to the load cells if mishandled.

3.3.2 Electromagnetic Force Restoration Scales

Electromagnetic force restoration scales use a magnet and coil system to generate a force that counteracts the weight of an object. The current required to balance this force is measured and converted into the corresponding weight. These scales are commonly used in pharmaceutical and scientific laboratories, medical research, and jewelry weighing due to their high precision. The primary advantage of this type of scale is its extremely high accuracy, even down to micrograms. Additionally, the lack of mechanical wear contributes to its reliability. However, these scales are very expensive and are sensitive to vibrations and environmental factors, making them unsuitable for portable use. They also have a limited weight capacity compared to other scales, though this is less of a concern for our design, as we are not dealing with heavy objects.

3.3.3 Capacitive Digital Scales

Capacitive digital scales measure changes in capacitance when weight is applied to a conductive plate. The force alters the dielectric properties, causing a shift in capacitance, which is then converted into an electrical signal representing the weight. Capacitive scales are commonly used for kitchen applications, postal services, and other compact weighing needs. They require fewer components, making them ideal for portable designs, and they consume minimal power. However, they are generally less accurate than other types of scales, and their reliability can be affected by temperature fluctuations.

3.3.4 Hydraulic or Pneumatic Scales

Hydraulic scales determine weight by measuring changes in fluid or air pressure. When weight is applied, the pressure increases, and this change is converted into a weight measurement. These scales are commonly used in construction, harsh environments, and for weighing extremely large objects. Their main advantage is their resistance to shock and overload, making them highly durable. Additionally, the absence of electrical components reduces safety risks in hazardous conditions. However, hydraulic scales are not suitable for portable designs, which is a key requirement for our project.

3.3.5 Piezoelectric Digital Scales

Piezoelectric digital scales utilize piezoelectric crystals, which generate an electrical charge when subjected to mechanical stress. The weight is determined by analyzing this charge output, as it is proportional to the applied force. These scales are commonly used in nanotechnology, aerospace, and medical applications, where extreme precision and stability are required. While their accuracy is a major advantage, they are impractical for our project due to their high cost and the constraints of our budget.

3.3.6 Smart and IoT Connected Scales

Smart scales use communication technologies like Bluetooth or Wi-Fi to send weight data to apps or connected software. They can integrate with Point of Sale (POS) systems to perform tasks such as syncing inventory, tracking health data, and logging information. Key advantages of smart scales include remote monitoring, data tracking, and integration with smart devices and automation, all of which enhance efficiency. However, they are typically more expensive than traditional digital scales and require ongoing software support, which can be a disadvantage.

Additionally, the need for internet connectivity can interfere with portability, making them less suitable for mobile use.

3.3.7 Scale Part Selection

The scale we have decided to build is the Strain Gauge Load Cell. One of the reasons we chose this scale architecture is because of its compatibility with the ESP32 microcontroller we will be using. Another reason is due to its low voltage which allows it to be easily integrated with battery powered systems. Lastly, the portability of this scale compared to the others was a huge deciding factor since it is a major requirement of our project.

3.4 Weight Sensors

Due to the specific design requirements of our project, we will be utilizing a load cell as our primary weight sensor. Load cells offer high accuracy, reliability, and ease of integration with microcontrollers, making them well-suited for our application. Rather than relying on mechanical or non-electrical weight measurement methods, we will incorporate the load cell directly into our PCB design. This approach will allow for a more compact, efficient, and streamlined system, ensuring precise weight measurements while maintaining the overall portability of our device. By implementing the load cell on a PCB, we can also optimize signal integrity, reduce noise, and improve overall system stability, leading to more consistent and accurate readings.

3.4.1 Load Cells

Load cells are essential components in weight measurement systems, converting force into an electrical signal that can be processed and displayed. They come in various types, each designed for specific applications based on factors such as weight capacity, accuracy, and structural integrity.

3.4.1.1 Single-Point Load Cells:

Single-point load cells are among the most commonly used in small-scale weighing applications due to their lightweight design and high accuracy. One of their most significant advantages is their ability to measure weight accurately even when the load is not perfectly centered, making them ideal for compact digital scales. These load cells are widely used in kitchen scales, postal scales, and laboratory balances.

A well-known model in this category is the SparkFun SEN-13329, which has a maximum capacity of 5 kg. Made of aluminum, this load cell has an output sensitivity of approximately 0.9 to 1.1 mV/V and requires an excitation voltage of 5V. Its lightweight nature, affordability, and ease of integration with a load cell amplifier make it a popular choice for precision weighing applications. However, its relatively low weight capacity limits its use to light objects. Another widely used single-point load cell is the CZL635, which has a capacity of 10 kg. Like the SparkFun model, it is also made of aluminum but offers a slightly broader output sensitivity range of 0.85 to 1.15 mV/V, with an excitation voltage between 5V and 10V. This model is preferred for laboratory balances and lightweight industrial applications due to its enhanced durability, though it still shares the limitation of being restricted to low-weight measurements.

3.4.1.2 Bar-Type Load Cells

For applications requiring higher weight capacities, bar-type load cells are often the preferred choice. These load cells are commonly used in medium-scale weighing systems such as luggage scales, bench-top industrial scales, and larger digital weighing platforms. Compared to single-point load cells, bar-type load cells are structurally more robust but require an evenly distributed load to maintain measurement accuracy.

A widely used model in this category is the HX711-Compatible 50 kg load cell, which is constructed from aluminum alloy and offers an output sensitivity between 0.85 and 1.15 mV/V. It operates with an excitation voltage of 5V to 10V and integrates seamlessly with the HX711 analog-to-digital converter (ADC) module, making it an affordable and accessible choice for digital platform scales. However, its primary drawback is its limited ability to handle uneven weight distribution. Another high-quality bar-type load cell is the Bosche H30A, which provides a maximum capacity of 60 kg and is constructed from stainless steel, giving it superior durability. This model features an output sensitivity of 2.0 mV/V and an excitation voltage range of 5V to 12V. While its stainless-steel build ensures longevity and resistance to environmental factors, it is more expensive than aluminum alternatives. The Vishay Tedeo-Huntleigh 1042 is another widely recognized bar-type load cell that is available in maximum capacities of 50 kg, 100 kg, and 200 kg. Made from stainless steel, it delivers an output sensitivity of 2.0 mV/V and operates within an excitation voltage range of 5V to 12V. Due to its high accuracy, it is commonly used in larger digital weighing systems and medical scales. However, precise mounting is crucial for optimal performance, as improper installation can impact measurement accuracy.

3.4.1.3 S-Type Load Cells

S-type load cells offer even higher weight capacities while maintaining precision, making them ideal for force measurement and industrial weighing applications. These load cells are distinguished by their "S" shape, which allows them to measure both tension and compression forces with high accuracy.

One notable model is the Vishay Tedeo-Huntleigh 614, available in maximum capacities of 100 kg, 200 kg, and 500 kg. Constructed from nickel-plated alloy steel, it has an output sensitivity of 3.0 mV/V and an excitation voltage range of 5V to 15V. Due to its durability and precision, it is widely used in force measurement systems and industrial applications where highly accurate load measurements are required. However, it is less commonly used for personal weighing devices due to its high weight capacity and structural complexity. Another reliable model in this category is the Zemic H3-C3-100kg-3B, which has a maximum capacity of 100 kg and is made of stainless steel. Operating with an excitation voltage of 5V to 15V, this load cell is commonly used for suspended weighing applications and dynamic force measurement. It features excellent overload protection, making it suitable for environments where unexpected force fluctuations may occur. However, its enhanced durability and advanced features make it more expensive than standard bar-type or single-point load cells.

3.4.1.4 Compression Load Cells

For large-scale industrial and heavy-duty weighing applications, compression load cells are the preferred choice. These load cells are specifically designed to measure compressive forces and

are used in applications such as truck scales, weighbridges, and bulk material measurement systems.

The Vishay Celtron CLB is a well-regarded compression load cell with maximum capacities ranging from 500 kg to 10,000 kg. Made of stainless steel, it offers an output sensitivity of 2.0 mV/V and operates within an excitation voltage range of 5V to 20V. This model is widely used in high-capacity industrial weighing systems where precise and reliable weight measurement is critical. While it provides exceptional durability and accuracy for large-scale applications, it is not suitable for portable digital scales due to its high weight capacity and structural bulk.

Each type of load cell serves a specific purpose, with single-point and bar-type load cells being the most appropriate for compact and portable digital scale designs. The choice of load cell ultimately depends on the weight capacity requirements, environmental conditions, and level of accuracy needed for a given application.

Table 4: Load Cell Comparison

Load Cell Type	Model	Max Capacity	Material	Output Sensitivity	Excitation Voltage	Key Features
Single-Point	SparkFun SEN-13329	10 kg	Aluminum	0.9 - 1.1 mV/V	5V	Lightweight, high accuracy, ideal for compact digital scales
Single-Point	CZL635	10 kg	Aluminum	0.85 - 1.15 mV/V	5V - 10V	Slightly higher capacity, used in laboratory and small industrial scales
Bar-Type	HX711 – Compatible 50kg Load Cell	50 kg	Aluminum Alloy	0.85 - 1.15 mV/V	5V - 10V	Affordable, pairs well with HX711 ADC, best for platform scales
Bar-Type	Bosche H30A	60 kg	Stainless Steel	2.0 mV/V	5V - 12V	Superior durability, used in bench-top industrial scales
Bar-Type	Vishay Teda – Huntleigh 1042	50 kg, 100 kg, 200 kg	Stainless Steel	2.0 mV/V	5V - 12V	High accuracy, commonly used in medical and industrial scales

S-Type	Vishay Teadea – Huntleigh 614	100 kg, 200 kg, 500 kg	Nickel-Plated Alloy Steel	3.0 mV/V	5V - 15V	Measures both tension and compression, used in force measurement systems
S-Type	Zemic H3-C3-100kg-3B	100 kg	Stainless Steel	3.0 mV/V	5V - 15V	High overload protection, ideal for suspended weighing applications
Compression	Vishay Celtron CLB	500 kg – 10,000 kg	Stainless Steel	2.0 mV/V	5V - 20V	Heavy-duty industrial applications, ideal for truck scales and bulk material measurement

3.4.1.5 Load Cell Part Selection

For our load cell selection, we have chosen the SparkFun SEN-13329 due to its compact, lightweight design and high accuracy, which is essential for precise breastmilk labeling. Its 5 kg maximum capacity makes it well-suited for small-scale applications like our digital scale. Additionally, its 5V excitation voltage is easily supported by our system, simplifying integration. The SEN-13329 is also affordable and readily available, making it an ideal choice for both prototyping and final implementation.

3.4.2 Load Cell Amplifier

In our design, a load cell amplifier is used to properly translate the small and often noisy signals from our load cells into a form that is readable by our microcontroller. These amplifiers are essential in providing signal conditioning, filtering, and amplification to ensure accurate and stable weight measurements. By carefully selecting the right amplifier, we have optimized the performance of our weighing system and minimize errors due to noise, temperature changes, and electrical interference.

3.4.2.1 HX711 Load Cell Amplifier

One of the most used load cell amplifiers is the HX711. It is a low-cost, 24-bit analog-to-digital converter (ADC) specifically designed for load cell applications. With built-in gain settings and a simple two-wire communication interface, the HX711 makes integration with microcontrollers straightforward. This amplifier is widely used in digital scales and small weighing systems due to its ease of use and affordability.

One well-known module featuring the HX711 is the SparkFun Load Cell Amplifier – HX711, which offers selectable gain settings of 32x, 64x, and 128x. It operates on both 5V and 3.3V power supplies, making it highly compatible with different microcontrollers, including the ESP32.

Another widely available module based on this amplifier is the DFRobot Gravity HX711, which features onboard screw terminals for easy wiring and connection to load cells. Additionally, there are numerous generic HX711 modules available from various manufacturers. These lower-cost alternatives can be useful for prototyping, but their quality and performance can vary, requiring additional calibration and testing.

3.4.2.2 ADS1232 Load Cell Amplifier

For applications requiring higher precision and better noise filtering, the ADS1232 is an excellent alternative. Manufactured by Texas Instruments, this 24-bit high-resolution ADC offers superior noise reduction and signal stability compared to the HX711. It includes onboard low-noise filtering and is designed to provide more accurate weight readings in environments where electrical interference could be an issue.

A closely related model, the ADS1234, shares the same core architecture but includes four differential input channels. This allows for multiple load cells to be connected simultaneously, making it ideal for applications where weight distribution needs to be measured at multiple points.

3.4.2.3 NAU7802 Load Cell Amplifier

Another high-precision load cell amplifier that stands out for its ease of integration is the NAU7802. Unlike the ADS1232, which uses an SPI communication protocol, the NAU7802 utilizes an I²C interface, simplifying communication with microcontrollers and allowing multiple sensors to be connected on the same data bus.

One of the most widely used modules featuring this amplifier is the SparkFun Qwiic NAU7802, which is designed for easy integration into embedded systems. The I²C communication capability makes this amplifier particularly suitable for applications where multiple weight sensors or other I²C peripherals, such as a touchscreen or additional sensor modules, need to operate in parallel.

3.4.2.4 Analog Load Cell Amplifier

For more custom weight measurement systems that require greater flexibility, analog load cell amplifiers can be used instead of dedicated digital ADC modules. These amplifiers take the small voltage signals generated by a load cell and amplify them before sending them to an external ADC for digitization.

One example of this type of amplifier is the Texas Instruments INA125, a high-precision instrumentation amplifier with a built-in voltage reference. Unlike dedicated ADC-based modules like the HX711 or NAU7802, the INA125 requires external circuitry for gain adjustment and signal conditioning. However, it offers greater customization and can be fine-tuned for applications requiring specific signal amplification characteristics.

Another similar option is the Analog Devices AD620, which also provides excellent gain control and stability for weight measurement applications but requires additional external components for proper operation.

The selection of the right load cell amplifier for our project depends on factors such as precision requirements, ease of integration, power consumption, and communication protocols. The HX711 is a reliable choice for low-cost, simple weighing systems, while the ADS1232 and NAU7802 offer superior accuracy for more demanding applications. Analog amplifiers such as the INA125 provide a more flexible but complex solution that can be tailored to specific measurement needs. By evaluating the trade-offs between these amplifiers, we can determine the best fit for our scale's performance and functionality.

Table 5: Load Cell Amplifier Comparison

Amplifier	Type	Resolution	Interface	Power Supply	Key Features	Common Modules
HX711	Digital ADC (24-bit)	24-bit	Two-wire	5V / 3.3V	Low-cost, built-in gain settings (32x, 64x, 128x), easy integration	SparkFun HX711, DFRobot Gravity HX711, Generic HX711 modules
ADS1232	Digital ADC (24-bit)	24-bit	SPI	5V	High precision, low-noise filtering, better stability than HX711	Texas Instruments ADS1232
ADS1234	Digital ADC (24-bit)	24-bit	SPI	5V	Similar to ADS1232, but supports multiple load cells (4 channels)	Texas Instruments ADS1234
NAU7802	Digital ADC (24-bit)	24-bit	I ² C	3.3V / 5V	High-precision, easy multi-sensor integration, low power	SparkFun Qwiic NAU7802

INA125	Analog Instrumentation Amp	N/A	Analog Output	2.7V - 15V	Customizable gain settings, external ADC required, built-in voltage reference	Texas Instruments INA125
AD620	Analog Instrumentation Amp	N/A	Analog Output	$\pm 2.3V$ - $\pm 18V$	High gain control, external components needed for full operation	Analog Devices AD620

3.4.2.5 Load Cell Amplifier Part Selection

The type of Load Cell Amplifier we have chosen is the HX711. We have chosen this one due to its affordability, ease of use and built-in features. One of the main reasons we chose it is due to its simple two-wire interface. This will make it easier to integrate with the ESP32 and does not require external ADC, which simplifies our design. It also has built in gain adjustability allowing it to be compatible with different load cells and optimizes accuracy.

Table 6: HX711 Part Comparison

Module	Supply Voltage	Current Consumption	Interface	Gain Settings	Max Data Rate	Operating Temperature	Other Features
SparkFunS EN-13879	2.7V-5.5V	~1.5mA	Two-wire (clock data)	32x, 64x, 128x	80 SPS	-40°C to 85°C	High-quality PCB, Stable Performance
DFRobot Gravity HX711	2.7V-5.5V	~1.5mA	Two-wire (clock data)	32x, 64x, 128x	80 SPS	-40°C to 85°C	Plug & Play with Gravity ecosystem
Adafruit HX711 Module	2.7V-5.5V	~1.5mA	Two-wire (clock data)	32x, 64x, 128x	80 SPS	-40°C to 85°C	High PCB Quality, Breadboard-friendly
Generic HX711 Module	2.6V-5.5V	~1.5mA	Two-wire (clock data)	32x, 64x, 128x	80 SPS	-40°C to 85°C	Varies in Quality, Low-cost Alternative

The load cell amplifier component we chose was the Spark-Fun SEN-13879. The main reason we chose this is because we also picked the SparkFun SEN-13329 as our load cell, which is directly compatible with this amplifier. Other reasons we chose this amplifier include reliability,

wide power supply range, and low power consumption. This amplifier is known for its high-quality PCB which ensures low noise and high stability. We were also able to find a lot of schematics and design examples for it. The amplifier is also compatible with our microcontroller (ESP32) and supports the 3.3V it uses. Lastly, it helps maintain power efficiency, which is important for our battery-operated device.

3.5 Printer

To complete our scale system and provide a tangible output of measurement results, we have integrated a printer into the overall architecture. This printer serves as the final stage in the data pipeline, receiving processed information from the microcontroller and producing printed labels that include critical information such as the measured weight of the breastmilk and any dietary notes or restrictions specified by the user. These printed labels are essential for caregivers and healthcare professionals who need quick, accurate, and physical records of feeding data for tracking and documentation purposes.

The printer communicates directly with the ESP32 microcontroller, which formats and sends the print data via a serial interface. This includes not only the numeric weight measurement, but also customizable text entries, such as allergy alerts, entered through the touchscreen interface or a connected mobile app.

We evaluated multiple printer options to determine the best fit for our system. Key criteria include print speed, resolution, power requirements, size, communication interface compatibility, and ease of integration with our existing hardware and firmware. Through careful selection and integration of the printer, our device is able to provide professional-quality printed labels that enhance both the user experience and the utility of the scale.

3.5.1 Zebra ZQ620 / ZQ520

The Zebra ZQ500 series Mobile Printers have a military-grade durability against various materials like concrete and sprayed liquids that could be useful for our product, as it will be thrown into on-the-go bags and must withstand potential bumps against other objects, as well as potential milk spills. The printer also comes with compatibility to an existing Zebra mobile app to configure different elements of the label printing, such as print quality and sizing. The ZQ500 series are all between 3-4 inches, making it compact enough to implement into our design. Despite these useful features, the Zebra ZQ620 sells for a relatively high price compared to our budget, with product price-points showing a range of >\$600.

3.5.2 Brother RJ-2035B

Brother advertises this RJ-2035B printer as being ultra-compact “yet extremely rugged with durable construction”. They claim an IP54 certification rating, protecting the printer from limited dust and water splashes in any direction. Other specs they advertise are Bluetooth, NFC quick connection, 4 ips (inches per second) print speed, colored LED indicators, support for multiple operating systems, and a long-lasting battery life (although the exact lifetime is unspecified). The selling points are very similar to the Zebra option, but the advantage of the Brother printer is that it retails for \$380 on the official Brother website – a nearly 40% drop in price.

3.5.3 Bixolon SPP-R310

The Bixolon printer boasts wireless and wired connectivity options including Bluetooth® (MFi certified), Dual-band WLAN (5 GHz and 2.4 GHz), USB and Serial. It also features long-lasting battery life (24hr standby time). It has the capability of printing both labels and receipts, with print speeds of 100mm/sec for receipts and 80mm/sec for labels. Lastly, it is a 385g lightweight design that is easy to grip. For our product, the various connectivity options give us more flexibility in label control, and like the previous printers the durability ratings would help in creating a product that can hold up with potential accidents. The price range is also more desirable for us, selling online in the range of \$250-\$400.

3.5.4 MUNBYN ITPP941

The MUNYUN printer deviates the most from the other printer designs with support for various label sizes, a tradeoff for its larger & bulkier size. Since the printer is made more for office use, it does not aim for the same compact use as the other printers, which is its biggest downside for our project. There is also no Bluetooth connectivity, and functions solely through wired connections. However, the biggest advantage is its affordability with a cost that is less than \$200.

Table 7: Printer Comparison

Printer	Price	Print Speed	Power Consumption / Battery Life	Connectivity	Label Size Supported
Zebra ZQ620/ZQ520	>\$600	4.5(114.3mm) inches per sec	Battery powered	Bluetooth, USB	3-4 inches
Brother RJ-2035B	\$380	4(101.6 mm) inches per sec	Long lasting battery	Bluetooth, NFC, USB	2 inches
Bixolon SPP-R310	\$250-\$400	80 mm/sec for labels	24-hr standby battery life	Bluetooth,WLAN, USB	3 inches
MUNBYN ITPP941	<\$200	150 mm/sec	Wired power, no battery	USB	Supports various label sizes
Primuz Micro Thermal Printer	\$46.88	Up to 60 mm/sec	Wired power, no battery	TTL Serial, RS232	57x30 mm

3.5.5 Printer Part Selection

For our printer selection we went with the Primuz Micro Thermal printer. While doing research for the printer the main issue we were running into is finding a thermal printer that can be integrated into our system and supports label printing. We found a lot of printers that would integrate into our system, but they didn't support label printing. The options we found that do support label printing have their own interface, which means we would have to deconstruct the printer or use the interface they already have. With the Primuz Micro Thermal Printer we were able to easily integrate it into our system because it has a power supply wire, serial interface, and supports the microcontroller we chose.

3.6 Display Screen

During our research, several options for display screens emerged as viable options. The most notable among them were the OLED, LCD, ePaper and Multi-Segment LED display screens. Each offers unique advantages in different areas such as power consumption, readability, and integration complexity.

3.6.1 OLED (Organic Light Emitting Diode)

The OLED display screen is a flat panel screen commonly used in TVs, smartphones, monitors, and tablets. It works by passing an electric current through organic materials, which in turn emit light. By eliminating the need for a backlight, it allows the device to be more lightweight. The organic material used not only emits colors but also allows for better contrast ratios. Additionally, OLED displays are more energy-efficient and require less power consumption due to the absence of a backlight.

There are two types of OLED displays, PMOLED (Passive Matrix Organic Light Emitting Diode) and AMOLED (Active Matrix Light Emitting Diode). The PMOLED display uses an external semiconductor to control which pixels are lit. As a result, the size is limited to around 320 x 160 pixels. On the other hand, the AMOLED display can offer different levels of colors at each pixel rather than being restricted to only a few levels, as seen in PMOLED displays.

There are many advantages to using the OLED display. One key benefit is that, since it generates light directly, it requires less power consumption. This is especially important because we are trying to create a portable, battery-efficient device. Using components with lower power consumption will contribute to longer-lasting battery life. It has several other advantages, its flexibility, thinness, and lightweight design. This makes it particularly ideal for portable devices since it requires fewer layers without compromising performance.

Furthermore, the OLED display screen can support a wide range of temperatures, from -40°C to 80°C. This is important because, as a portable device, we don't know what climate the device will be used in. All components must withstand temperature fluctuations because they may be in different environments, like a hot car or cold hospital freezer. The benefit of the display screen being able to handle this will prevent it from freezing or becoming unresponsive at low temperatures and screen degradation at high temperatures. One disadvantage that we must consider is that OLED screens are more expensive. Since we are working within a budget, it may not be cost-effective to buy these in bulk. Another disadvantage of using this type of display is the fact the organic material's lifespan is shorter than other display screens. A variety of factors can lead to the quickened degradation of the display screen including constant high temperatures.

3.6.2 LCD (Liquid Crystal Display)

An LCD is a display screen that uses liquid crystals to modulate light. The screen utilizes two glass plates and sandwiches the liquid crystals in between. The liquid crystals are arranged in specific patterns that allow them to be manipulated when different electric currents are applied. This manipulation affects how the light passes through, which in turn allows for different images to appear.

There are three different types of LCD display screens: TN (twisted nematic), IPS (in-plane switching), and VA (vertical alignment). Twisted nematic is most commonly found one in LCDs. The liquid crystals twist at 90 degrees when electricity is applied. They are low cost and have high response times but have limited viewing angles and poor color accuracy compared to the other types. IPS works by having the liquid crystals rotate parallel to let light in when voltage is applied to enhance the image quality of the screens. Some advantages of using this type are that it is cost effective, has a wide range of viewing angles, and color accuracy. One disadvantage is that it has a slightly slower response time in comparison to its counterparts, which is something we are focusing on diminishing. The last type is VA which has the liquid crystals align vertically with the glass allowing light to pass through when voltage is applied. Some of the pros include wider viewing angles and improved color/contrast compared to TN. The biggest disadvantage is that it has higher power consumption, something we actively want to avoid.

The overall advantages of using an LCD display include low power consumption, thinness, and lightweight design. All these aspects are important since, as mentioned earlier, our device is portable and therefore, we want it to have a long-lasting battery life. Another benefit is that it has good color representation and accuracy on the screen itself. However, LCD displays also have drawbacks. Some disadvantages to using LCD displays are limited viewing angles, slower response time, the requirement of a backlight, and lower contrast than other displays. One of our main concerns with this display is its response time, as we want to minimize latency, which may not be as feasible with an LCD. Additionally, the need for a backlight means more power consumption and adds weight, which is not ideal for our design requirements.

3.6.3 Electronic Paper Display

An ePaper display is a type of screen created to replicate the look and feel of printed paper. This technology is not only energy efficient but also has minimal blue light emission. It functions by utilizing microcapsules filled with charged black and white particles. By applying an electric field, it moves the particles to form text and images. Unlike the LCD it does not require a backlight to do so.

There are three different kinds of ePaper displays: Monochrome, Color, and Flexible. As their names suggest, the first two types represent whether it is able to produce color or not. Monochrome produces black and white displays that need high contrast and low power usage, in devices such as kindles. While color ePaper displays are not as vivid in color as LCD or OLED displays, they are commonly used for digital advertising where power is more of a concern than color variety. The last type, Flexible ePaper, is built on bendable materials that allows it to be more adaptable for various applications such as wearable medical devices.

The biggest advantage of using this type of display would be the fact that it has extremely low power consumption. The only time it uses power is to update the display. Other notable advantages of using this display are that it does not have any glare from the sunlight, does not emit blue light, and has a wide range of viewing angles. The cons of the device heavily outweigh its benefits, due to the fact that its touchscreen capabilities are very limited. A major hardware component in our project design is a touchscreen display that creates an interactive user interface on the device. With the ePaper display, it can support capacitive touch but has a very slow

refresh rate making it unsuitable for interactive applications. Other disadvantages of this device include its limits in color range, fragility, and cost when compared to LCD displays.

3.6.4 Multi-Segment LED Displays

Multi-Segment LED Displays are seen very often in everyday life in digital clocks, scales, calculators, and more. The display consists of light emitting diodes (LED) arranged in segments that light up in different combinations to create numbers, letters, and symbols. Each segment has its own circuit that can be turned on and off using a driver IC or microcontroller.

There are four different types of Multi-segment LED displays: 7-Segment, 14-Segment, 16-Segment, and Dot Matrix. The 7-Segment display consists of 7 individual segments arranged in the shape of an 8 with an optional decimal point. This type can display numbers 0-9 and some letters. Some benefits of this type of display are that it's easy to interface with a microcontroller, low cost, low power consumption, and long lifespan. This type is limited in what can be displayed and has low resolution. The difference between the 14-Segment and the 16-Segment is the 14 can support both letters and numbers, but the 16 allows them to be more legible. As you go up in the number of segments it means higher power consumption as well as more complex circuitry. The Dot Matrix has a different setup where the LEDs are not put into segments but instead grids to be able to display any character, number, or symbol. This type can support scrolling text, animations, and different font styles. It would be most suitable for the display we need since it is the most flexible with its functionality and can be expanded by linking multiple displays together. This type of display does require more complex control and consumes more power than the other three types.

The two biggest advantages of using the multi-segment display would be that it is extremely power efficient and has a long lifespan in comparison to the OLED and LCD. However, it is very limited in display capabilities, has no multi-touch support at all, and can't display interactive elements.

Table 8: Display Screen Comparisons

Display Type	Power Consumption	Viewing Angles	Refresh Rate	Color	Durability	Cost	Touchscreen Support
OLED	High	Excellent	Very Fast	Full Color	Moderate (Burn-in-risk)	\$50-\$200	Excellent
LCD	Medium	Good	Moderate	Full Color	High	\$30-\$100	Excellent
EPaper	Ultra Low	Excellent	Very Slow	Limited	Fragile	\$15-\$50	Limited
Multi-Segment LED	Very Low	Limited	Limited	Monochrome or some color	Very High	\$5-\$40	None

3.6.5 LCD Part Selection

After doing research on possible display screens we could use for our project, we decided on using an LCD touchscreen display. Out of the options we mentioned only LCD and OLED were able to support touchscreen capabilities, which is a huge factor in our project. The reason why we decided on the LCD specifically is that it is more affordable, has no significant burn-in risk like the OLED, and we have prior experience using LCDs. Since the touchscreen serves as the main input and output interface for our device, it's essential that the display is both reliable and user-friendly. We also determined that our screen should be about 4 to 5 inches, since there are a lot of options on the screen, but it needs to be small enough to fit within our expected dimensions. The options we have decided to proceed with are the ILI9488 LCD Display, DT050BTFT – 5" Color TFT LCD Display, and NHD-4.3-480272FT-CSXP-T (4.3" TFT LCD Display).

3.6.5.1 ILI9488 TFT Display

The ILI9488 is a powerful TFT (Thin-Film Transistor) LCD controller commonly used to drive color displays, typically in the 3.5-inch to 4-inch range, with a standard resolution of 320×480 pixels. Displays using the ILI9488 controller are known for their vivid color reproduction, high contrast, and sharp image quality, making them a strong choice for applications that require a clear and responsive visual interface.

One of the key advantages of the ILI9488 is its flexible interface options. It supports SPI for easy integration with microcontrollers that have limited GPIO pins, as well as 16-bit parallel and RGB interfaces for higher data throughput when faster screen refresh rates are needed. This versatility allows developers to optimize the balance between performance and pin usage depending on the needs of the project.

Many ILI9488 displays come equipped with a resistive touchscreen, which can be operated with a finger or a stylus, offering reliable input even in situations where capacitive touchscreens may struggle. The resistive touchscreen typically includes a stylus for precise interaction, which can be helpful in navigating smaller UI elements.

The ILI9488 is well-supported by the ESP32, the microcontroller used in our project. Libraries such as TFT_eSPI make integration straightforward, enabling fast development and reliable communication between the microcontroller and the display. This compatibility allows us to fully utilize the ESP32's graphical and processing capabilities to deliver a polished, interactive user experience.

3.6.5.2 DT050BTFT

The DT050BTFT is a 5.0-inch color TFT LCD module developed by Displaytech, designed for high-quality graphical display in embedded systems. It features a resolution of 800×480 pixels, offering a sharp and detailed display that is well-suited for applications requiring clear visual output. The screen supports rich color depth, delivering vibrant and accurate color representation across a wide viewing area.

One of the standout qualities of the DT050BTFT is its clarity and responsiveness, which make it ideal for user interfaces that require high visibility and reliable performance. However, one limitation of this model is that it supports fewer interface options compared to some other displays. It primarily uses a 24-bit RGB parallel interface, which provides fast data transfer and smooth graphics rendering, but may require additional pins and setup on certain microcontroller platforms.

The module includes an integrated white LED backlight, ensuring good visibility even in well-lit environments. For projects such as ours requiring touch functionality, the DT050BTFT-PTS variant includes a capacitive touchscreen with an I²C interface, allowing for smooth, multi-touch input and easy integration with modern microcontrollers such as the ESP32. The screen is large enough to display detailed menus and interactive controls, yet still small enough to fit within the physical constraints of a handheld or benchtop unit, which is perfect for our project's needs.

3.6.5.3 NHD-4.3-480272FT-CSXP-T

The NHD-4.3-480272FT-CSXP-T is a 4.3-inch TFT LCD module developed by Newhaven Display, designed for embedded applications that require a compact yet high-quality display. It features a 480 × 272-pixel resolution (WQVGA), delivering detailed visuals that are ideal for user interfaces, data visualization, and real-time system feedback. This display incorporates a resistive touchscreen, making it highly versatile in various operating environments. The resistive technology also offers reliability and accuracy for applications where single-point touch is sufficient.

One of the key advantages of this module is its flexible communication interface. It supports multiple SPI modes, including standard SPI, dual SPI (D-SPI), and quad SPI (Q-SPI), giving developers a range of options when connecting to microcontrollers or development boards. This flexibility makes it well-suited for integration into systems with limited GPIO resources or those requiring faster data transfer.

The NHD-4.3-480272FT-CSXP-T operates at a 3.3V supply voltage, which is compatible with most modern microcontrollers such as the ESP32, minimizing the need for additional voltage regulation components. Its size also makes it a strong candidate for portable devices, providing enough screen space for intuitive navigation while maintaining a compact footprint.

Table 9: LCD Display Screen Comparison

	ILI9488 TFT Display	DT050BTFT – 5" Color TFT LCD Display	NHD-4.3-480272FT-CSXP-T (4.3" TFT LCD Display)
Screen Size	3.5 inches	5.0 inches	5.0 inches
Resolution	320 x 480	800 x 480	480x272
Display Type	TFT LCD	IPS TFT LCD	IPS TFT LCD
Viewing Angle	Wide-angle	ALL(IPS)	ALL(IPS)
LCD Driver IC	ILI9488	ILI6137	ILI6137

Interface	Multiple including SPI	SPI/RGB	SPI
Supply Voltage	3.3V - 5V	3.0-3.6V	3.0-3.6V
Operating Temperature	~ -20-60 °C	~20-70 °C	~20-70 °C
Touchscreen options	Resistive	Resistive or Capacitive	Resistive
Cost	\$16.99	\$55-\$92	\$60-\$75

We chose the ILI9488 TFT Display for its cost-effectiveness while still offering a comparable resolution and high visibility. Among the options considered, it is the most affordable, making it a practical choice without compromising essential features. Additionally, its 3.3V - 5V supply voltage aligns with the other models, eliminating the need for extra voltage regulators. The SPI interface ensures seamless integration with the microcontrollers we used, further simplifying the design.

3.7 LEDs

In our project, we have implemented visual feedback using LEDs to provide important status indications. LEDs serve as a user-friendly way to communicate the device's state at a glance. Specifically, we will include 3 LEDs to indicate that the device is powered on, connected to WiFi, & connected to Bluetooth. These LEDs will enhance usability by giving clear, real-time feedback without requiring a display or serial output.

3.7.1 Standard Through-Hole LEDs

Standard through-hole LEDs are simple, low-cost, and easy to integrate into our circuit. They typically come in a 3mm or 5mm package and require a current-limiting resistor, typically between 220Ω and 1kΩ, to prevent damage from excessive current. The most common colors for status indication include red, which is commonly used for low battery or error signals, green, which is often used for power-on indicators, and blue, which is sometimes used for general status indication. Through-hole LEDs can be directly connected to a microcontroller GPIO pin, allowing us to turn them on or off programmatically based on battery status or power conditions. Their main advantages are their simplicity, durability, and ease of prototyping, making them ideal for initial breadboard testing and later integration into a PCB.

3.7.2 Surface-Mount Device LEDs

Surface-mount device LEDs are compact, lightweight, and well-suited for PCB integration. They come in different sizes, such as 0805 or 1206, making them ideal for small-scale or space-constrained designs. These LEDs can be placed directly on the PCB near the battery management circuit, reducing wiring complexity and improving reliability. However, SMD LEDs require precise soldering techniques, making them slightly more challenging to work with compared to through-hole LEDs. They also need careful PCB layout planning to ensure proper current-limiting resistor placement and correct routing to the microcontroller. Despite this, they are ideal for final production designs due to their compact nature and high efficiency.

3.7.3 Bi-color or Tri-color LEDs

Bi-color and tri-color LEDs are multi-functional LEDs that can display different colors based on the voltage applied. These LEDs typically have a common anode or common cathode configuration, meaning they share a common pin while allowing independent control of different color segments. A bi-color LED can display red to indicate low battery and green to indicate sufficient power. Similarly, a tri-color LED can offer additional status signals, such as red for low battery, green for normal operation, and blue for charging. These LEDs are efficient for space-saving designs because they replace multiple separate LEDs with a single component. However, they require careful microcontroller control, as the correct voltages must be applied to achieve the desired colors.

3.7.4 WS2812B

WS2812B LEDs are programmable, addressable RGB LEDs that allow for dynamic status indication. Unlike standard LEDs, which require a separate GPIO pin for each LED, WS2812B LEDs can be controlled using a single microcontroller pin via a serial data protocol. These LEDs are beneficial for applications where more advanced feedback is needed. Instead of a single-color low battery warning, a WS2812B LED could gradually change from green to yellow as the battery discharges, flash red when the battery is critically low, and pulse blue while charging before turning solid green when fully charged. While these LEDs provide highly customizable feedback, they are more complex to integrate and may be considered unnecessary for basic indicators. However, they are ideal if we want to provide a more visually appealing and user-friendly battery level display.

For our project, the final choice of LED will depend on factors such as simplicity, power efficiency, and ease of integration. If we want a straightforward and reliable approach, standard through-hole or SMD LEDs will suffice. If we prefer a more compact and feature-rich solution, bi-color LEDs may be the best option. If we want to add dynamic visual feedback, WS2812B LEDs would be the most suitable.

Table 10: LEDs Comparison

LED Type	Size/Form Factor	Power Consumption	Control Method	Advantages	Disadvantages
Standard Through-Hole LEDs	3mm, 5mm (Through-Hole)	Low (Requires resistor)	Direct GPIO control	Simple, low-cost, easy to prototype	Requires external resistors, larger footprint
Surface-Mount Device (SMD) LEDs	0805, 1206 (SMD)	Low (Requires resistor)	Direct GPIO control	Compact, ideal for PCB integration, efficient	Requires precise soldering, harder to replace
Bi-color LEDs	Common Anode/Cathode	Low (Depends on active color)	Direct GPIO control	Displays two colors with one LED, space-saving	Requires multiple control signals,

					limited color options
Tri-color LEDs	Common Anode/Cathode	Low (Depends on active color)	Direct GPIO control	Displays three colors, flexible status indication	Requires more complex control logic
WS2812B (Addressable RGB LEDs)	Varies (SMD/Through-Hole)	Moderate to High	Serial data (1-wire)	Fully programmable, dynamic color transitions, requires only one microcontroller pin	More complex integration, higher power consumption

3.7.5 LED Part Selection

For our LED selection, we used standard through-hole LEDs due to their ease of use and direct compatibility with microcontroller GPIO pins when paired with a current-limiting resistor. They allow for quick testing and modifications, making them ideal for initial development.

3.8 Power Supply

Based on our calculations, the power output of the device is estimated to range between 28.17905 W and 73.7325 W. This range will guide our selection of an appropriate power supply, ensuring that it can reliably support the system's minimum and maximum power demands while maintaining efficiency and stability.

Table 11: Power Usage Table

Component	Power Requirement	Power Usage
ESP32	2.2V-3.6V, 250mA	0.55W-0.9W
Printer	9V-24V, 3A	27W-72W
Load Cell Amplifier	2.7V-5V, <1.5mA	4.05mW-7.5mW
Touchscreen	2.5V-3.3V, 250mA	625mW-825mW

3.8.1 Primary (Non-Rechargeable) Batteries

Batteries are one of the most widely used power sources for portable devices, embedded systems and manufacturing applications. They can be split into two categories, primary and secondary. Primary batteries are ready to use and better for devices that need low power over long periods of time. They have a long shelf life, can be easily replaced, and are less likely to overheat/explode. Common types of batteries include alkaline, lithium, zinc carbon and silver oxide. For this project we will not be considering lithium batteries since they are highly flammable.

3.8.2 Alkaline Batteries

Alkaline batteries are commonly found in households powering toys, remote controls and other devices. They are known as your AA, AAA and 9V batteries which are easily found in stores for pretty cheap. They have a typical voltage output of 1.5V per cell for all batteries except the 9V rectangular ones. They have a high energy density ranging from 100-150 Wh/kg and long shelf life but low lifespan.

3.8.3 Zinc-Carbon Batteries

Zinc carbon batteries, however, are less common but still used for low-power devices such as clocks and have a lower energy density. They have an output voltage of 1.5V per cell and an energy density of 35-55 Wh/kg. They are known as your AA, AAA, C and D batteries.

3.8.4 Silver Oxide Batteries

Lastly are silver oxide batteries which are small button cell batteries commonly found in watches and hearing aids. They output a voltage of about 1.55V per cell. They are known for their high energy-density ranging from 100-180 Wh/kg, stable voltage output and long shelf life.

Some of the advantages of using primary batteries are, as mentioned before, that they have a long shelf life due to their slow discharge rate. They only use about 2-3% of their charge per year and can last up to 5 years or more. Another advantage is that they have a stable output meaning they are able to provide steady voltage throughout their lifespan. This also allows them to have longer runtimes since they have more energy stored per unit weight. Other reasons why we think these could work as our supply include the fact that they can operate in extreme temperatures, are readily available, and have a lost cost.

Although the advantages would be beneficial to our device, the drawbacks are just as significant too. Primary batteries are known for their longer lifespan but once they are depleted, they must be disposed of and replaced with new ones. This can be considered inconvenient in the sense that it is for mothers who are breastfeeding and may not have batteries at their disposal. This is also not economically friendly since it is contributing to electronic waste. Some of the metals the batteries are composed of can be toxic and harmful to dispose of. Another disadvantage is that even though they are cheaper in terms of cost upfront, over time they are generally less cost-effective due to the fact that they must be replaced. Another major disadvantage is that they are not sustainable for high drain devices such as power tools. This is important because our printer can use a lot of power, and we want to make sure our power supply supports the device's needs.

3.8.5 Secondary (Rechargeable) Batteries

Secondary batteries are mainly used for high-drain devices since they can be recharged and reused. They use an electric current to reverse the chemical reaction inside the batteries. Commonly used secondary batteries include Lithium-Ion, Lead-acid and Nickel metal hydride.

3.8.6 Lithium-Ion Batteries

Lithium-ion (Li-ion) batteries are widely used in portable electronics, electric vehicles, and other high-performance applications due to their high energy density (150-250 Wh/kg) and lightweight design. They typically output 3.6-3.7V per cell and have a relatively low self-discharge rate.

compared to other secondary batteries. One of the biggest advantages of this type of battery is their efficiency and long lifespan before noticeable degradation. They also have a high charge retention rate meaning they lose less energy when not in use. Although it does have some strong advantages, Li-ion batteries come with safety concerns. They are very sensitive to temperature and physical damage. This can lead to overheating, swelling or thermal runaway that may cause fires or explosions.

3.8.7 Lead-Acid Batteries

Lead-acid batteries are usually found in things like cars and backup power systems. Their output voltage is typically around 2V per cell, they come in 6V or 12V packs, and have an energy density of 20-50 Wh/kg. They are inexpensive, reliable, and have a high-power output. They are also heavy and bulky, which is not what we want for our lightweight portable device.

3.8.8 Nickle-Metal Hydride Batteries

Nickle metal hydride batteries are much safer than lithium since they have no expulsion risk, but they have a higher discharge rate (~1% per day). They output about 1.2V per cell with an energy density of 60-120 Wh/kg. They are often found in hybrid cars and as rechargeable AA/AAA batteries.

One major advantage of using a secondary battery is the fact that it is rechargeable which reduces waste and long-term cost. They also hold more power than primary batteries allowing for longer usage between charge cycles. The batteries are safer than primary batteries due to the fact that they will not leak harmful chemicals once degraded.

The rechargeable factor in secondary batteries is useful but leads to their biggest disadvantage, which is a higher energy loss each time it is charging and even when not in use. The lifespan is also very dependent on how many charging cycles it can handle before it needs to be replaced. There is also a safety concern with using them due to some being highly flammable or having the capability to explode if mishandled.

3.8.9 Solar Power

Solar Power is the process of converting sunlight into electricity using photovoltaic cells and other solar energy technologies. It is a renewable, clean energy source that can be used for portable devices, homes, businesses, and even large-scale power grids. Solar power requires 3 components which include solar panels, charge controller, and an inverter. There are three types of solar panels that can be used. Monocrystalline Solar Panels have the highest efficiency (~22%), a long lifespan, and work in conditions with dimmer light but are more expensive. Polycrystalline Solar Panels are more affordable with good efficiency (~16-18%) but require a larger size to produce the same power output as Monocrystalline. The last type is Thin-Film Solar Panels (~10-12%) which are lightweight and flexible, ideal for the pumped device and works well in high temperatures. It has a shorter lifespan and lower power efficiency than the other two types.

One major advantage to using solar panels is the fact that it is eco-friendly. There are no bounds to what it can power and requires minimal upkeep. However, it can be very costly just for the

solar panel itself and is dependent on sunlight. If it is too cloudy or raining the device may not function making it less reliable for mothers who intend to use it.

3.8.10 Supercapacitors

Supercapacitors are energy storage devices that store electrical energy through electrostatic fields rather than chemical reactions. They can discharge and charge quickly despite their size, making them ideal for things like flashlights or power backups. Their one major advantage is that, unlike batteries, their life cycle is extremely long, and it can last many cycles without significant degradation. It is also able to provide quick and powerful surges of energy for short periods, and they have better thermal stability compared to batteries. Some disadvantages to using a supercapacitor would be the gradual voltage decrease because it can make it harder to regulate a constant voltage. Another disadvantage is that it may be larger in size to store larger amounts of energy as a battery. Their energy density is typically only 5-15 Wh/kg which is significantly lower than batteries. This is something for us to consider due to the fact that we are trying to make more of a compact portable device.

Table 12: Power Supply Comparison

Power supply	Energy Density (Wh/kg)	Lifespan	Charge Time	Cost Range (\$)	Size and Portability	Environmental impact	Best For	Drawbacks
Alkaline Batteries (primary)	100-150	5+ years (shelf life)	Non-rechargeable	\$0.50-\$5 per battery	Compact and lightweight	E-waste (single use)	Low-power devices	Short lifespan, costly long-term
Zinc Carbon batteries (primary)	35-55	3-5 years (shelf life)	Non-rechargeable	\$0.30-\$3 per battery	Compact and lightweight	E-waste (single use)	Very low-power devices	Low energy output
Silver Oxide batteries (primary)	100-180	3-10 years	Non-rechargeable	\$1-\$10 per battery	Very small and light weight	More sustainable than alkaline	Watches and small electronics	Expensive per unit
Lithium-Ion Batteries (secondary)	150-250	500-2000 cycles	1-3hrs	\$30-\$150 per battery	Compact, varies by form factor	Recyclable	Electric vehicles and portable electronics	High cost, may be limited by charge/discharge overtime

Nickle Metal Hydride batteries (secondary)	60-120	500-1000 cycles	1-4 hrs	\$5-\$20 per battery	Medium weight, AA/AA A size	Rechargeable but self-charges quickly	Rechargeable household appliances	Lower power capacity than Li-ion
Lead-Acid batteries (secondary)	30-50	300-500 cycles	5-10 hrs	\$20-\$200	Large and heavy	Toxic chemicals, recyclable	High power applications	Bulk, not portable
Solar Power	Varies	20+ years	Dependent on sunlight	\$50-\$500	Panels can be compact	Sustainable and renewable	Outdoor/portable use	Sunlight dependent and high upfront cost
Supercapacitors	5-15	1,000,000+ cycles	Seconds	\$10-\$200	Small to medium	Recyclable, long-life span	Quick energy burst	Lower energy storage than batteries

3.8.11 Battery Part Selection

For our device we decided to go with the lithium-ion battery due to it providing an optimal balance of power, size and weight. The battery is rechargeable as well, which is the most feasible option for our portable device. Due to us having multiple components that may require different amounts of power, we wanted a battery that could support high energy density and be flexible when it comes to different power needs.

3.8.11.1 18650 Li-ion Battery

The 18650 Li-ion battery is a widely used rechargeable lithium-ion cell, named after its physical dimensions: 18mm in diameter and 65mm in length. It features a cylindrical form factor and is highly regarded for its balance of compact size, high energy density, and long-term reliability. This makes it a popular choice in applications ranging from consumer electronics to DIY projects, power tools, and even electric vehicles.

One of the key strengths of the 18650 battery is its energy efficiency. It typically delivers a nominal voltage of 3.6V to 3.7V, and when fully charged, can reach 4.2V. The cut-off voltage, which is the minimum safe operating level before the battery should be recharged, usually falls between 2.5V and 3.0V. These batteries are available in various capacities, commonly ranging from 1800mAh to over 3500mAh, allowing for flexibility depending on the power requirements of the application.

The 18650 also offers a long cycle life, usually rated for 300 to 500 charge-discharge cycles, with higher-end models achieving even more under proper usage conditions. It delivers consistent voltage output, which is critical for devices that rely on stable power, and it is both cost-effective and widely available. Overall, its durability, affordability, and versatility make it an excellent choice for portable electronics, including our project, where dependable, long-lasting power is essential.

3.8.11.2 14500 Li-ion Battery

The 14500 battery is a compact, rechargeable lithium-ion cell with the same physical dimensions as a standard AA battery, which is 14mm in diameter and 50mm in length. Although it resembles a typical AA battery in size, it operates at a much higher voltage, which makes it incompatible with standard AA battery-powered devices unless they are specifically designed to support lithium-ion cells.

The 14500 battery typically has a nominal voltage of 3.6V to 3.7V, and when fully charged, it reaches up to 4.2V. Its cut-off voltage is usually around 2.5V to 3.0V. In terms of capacity, it generally ranges from 600mAh to 1000mAh, which is lower than larger lithium-ion batteries like the 18650 but still sufficient for many low-power or space-constrained applications. Most 14500 batteries offer a recharge cycle life of around 300 to 500 cycles, depending on usage and charging practices.

One of the main advantages of the 14500 battery is its compact size, making it ideal for portable electronics, DIY projects, and devices where space is limited. It is lightweight and convenient, providing a good balance between power and portability. However, a notable limitation of the 14500 is its lower capacity and current output, which makes it unsuitable for high-drain devices, such as label printers or large display modules. These applications typically require more sustained power than a 14500 can reliably deliver.

In the context of our project, while the 14500 could potentially power low-energy components, it would not be ideal for high-power components like our touchscreen or label printer. For those parts, a battery with higher capacity and current output, such as the 18650, would be a more appropriate choice.

3.8.11.3 16340 Li-ion Battery

The 16340 Li-ion battery is a compact cylindrical rechargeable cell measuring 16 mm in diameter and 34 mm in length, hence the name 16340. It operates at a nominal voltage of 3.6V to 3.7V, with a fully charged voltage of 4.2V. The battery typically offers a capacity ranging from 600mAh to 1000mAh and has an average cycle life of approximately 300 to 500 charge-discharge cycles, depending on usage and quality.

While its small size makes it convenient for compact devices, the main limitation of the 16340 battery is its relatively low capacity, which results in shorter runtimes, especially in power-demanding applications. Additionally, it generally cannot support high current draw, making it unsuitable for systems that require sudden bursts of power or sustained high load, such as wireless communication modules or thermal printers. These limitations make it a less-than-ideal

choice for projects that involve multiple active components or peripherals, particularly where longer battery life and stable performance under load are critical.

Table 13: Lithium-Ion Battery Type Comparison

	18650 Li-ion Battery	14500 Li-ion Battery	16340 Li-ion Battery
Normal Capacity	1800 mAh – 3500 mAh	600 mAh – 1000 mAh	600mAh - 1000 mAh
Nominal Voltage	3.6V-3.7 V	3.6V-3.7 V	3.6V-3.7 V
Charge Voltage	4.2V	4.2V	4.2V
Weight	~50 g	~20 g	~20 g
Operating Temperature Charge	0-45 °C	-20-45 °C	-0-45 °C
Operating Temperature Discharge	-10-60 °C	-20-60 °C	-10-60 °C
Cost	\$3-\$5 (cheaper in bulk)	\$5-\$16	\$4-\$10

We have decided to go with the 18650 Li-ion battery for our project. One reason is due to the higher capacity than the other two batteries listed which allows us more flexibility since we do not yet know how much power will be required for our device. It also allows for a more extended runtime as well as reducing the need for persistent charging. This type of battery is also very commonly used in small electronics like laptops and flashlights.

3.8.12 Part Selection Battery Charging IC/Module

We have selected the MCP73837 as our battery charging module due to its versatility and suitability for our portable design. This module supports charging from both USB and an AC adapter, providing flexibility in power sources. Its compact form factor makes it ideal for our portable application, while its integrated protection features, including overvoltage, overcurrent, and thermal regulation, enhance reliability and safety. Additionally, the programmable charge current allows for customization based on battery capacity, and the built-in status indicators provide real-time charging feedback.

In our final design, instead of using an onboard voltage regulator IC to supply power, we opted for a portable battery pack to simplify the power system and increase mobility. After evaluating several options, we selected the TalentCell Rechargeable Battery Pack. This unit offers dual voltage outputs, 12V at 6000mAh and 5V at 12000mAh, which align well with the requirements of our system components, including the microcontroller, load cell amplifier, and touchscreen interface.

The TalentCell pack provided multiple advantages: it reduced the complexity of power distribution on our PCB, offered clean and regulated outputs without the need for additional circuitry, and ensured reliable operation during extended use. The inclusion of a 12.6V charger

made recharging straightforward, while its portability allowed the system to operate independently of a wall outlet, crucial for our goal of a fully portable design. Given its capacity and dual-output flexibility, this battery pack met both our technical and practical needs, making it an ideal choice for the final product.

Table 14: Battery Charging IC/ Module Comparison

	TP4056	BQ24296	TP5000	MCP73837
Best For	Single cell charging	Fast charging and USB-C PD	1S or 2S battery packs	Portable devices
Input Voltage	5V (Micro-USB or Type-C)	5V –17V(USB-C)	4V-9V	4.5V to 6V
Output Voltage	4.2 V	4.2 V	4.2 V (1S)/8.4 V(2S)	4.2V
Max Charge Current	1A	3A	1A	1A
Protection	Overcharge, Over-discharge and short circuit using DW01A and FS8205A MOSFETs	Thermal, Overvoltage and battery monitoring	Thermal and Overvoltage	Overvoltage, overcurrent, and thermal protection. Safety timer
Efficiency	~80-90%	~90-95%	~85-92%	~85-90%
Size	Compact	Small	Small	Compact
Cost	\$1-\$6	\$4-\$20	\$2-\$5	\$1.50-\$3.50

3.9 Voltage Regulators

In our project, we utilized voltage regulators to ensure that each component received the correct input voltage without overloading or underloading. Since our design includes multiple components with varying voltage requirements, such as the Arduino, touchscreen display, load cell amplifier, and label printer, proper voltage regulation is essential for stable and efficient operation.

Voltage regulators can be classified into two main types:

1. Linear regulators (LDOs) – Provide a stable voltage but are less efficient due to heat dissipation.
2. Switching regulators (buck/boost converters) – Efficiently step voltage up or down using high-frequency switching and external components like inductors and capacitors.

For our project, switching regulators are preferred due to their higher efficiency, making them ideal for battery-powered and portable applications. In this section we will discuss which regulators can be used in our project.

3.9.1 LM2576

The LM2576 is a step-down switching regulator that can take an input voltage of 7V-40V and output a fixed voltage of 3.3V, 5V, or 12V, or an adjustable voltage from 1.23V-37V. It has a

maximum output current of 3A and an efficiency of around 77-80%. While it includes built-in thermal shutdown and short-circuit protection, it requires external inductors and capacitors for operation. The LM2576 is best suited for simple, low-current applications where efficiency is important but high power is not required. In our project, it could be used for stepping down a high voltage source to 3.3V to power the HX711 load cell amplifier, although its lower efficiency compared to newer regulators makes it less ideal for high-power components like the label printer.

3.9.2 LM2679

The LM2679 is a high-current step-down switching regulator that supports an input voltage range of 8V-40V and offers output voltage options of 3.3V, 5V, 12V, or an adjustable range of 1.2V-37V. It has a maximum output current of 5A and a higher efficiency of around 85%. Operating at 260 kHz, its increased switching frequency allows for smaller external components and improved performance. Given its ability to supply higher current output, this regulator is well-suited for powering the label printer, which will likely require a stable 5A output.

3.9.3 LM2596

The LM2596 is another step-down buck regulator similar to the LM2576 but commonly found in ready-to-use modules, making it popular in DIY projects. It has an input voltage range of 4.5V-40V, an output voltage range of 1.23V-37V (fixed at 3.3V, 5V, or 12V), and a maximum output current of 3A. With an efficiency of 80-85%, it is a practical choice for stepping down voltage for microcontrollers. In our project, it is an excellent option for supplying 3.3V to the ESP32 and touchscreen display, offering a compact and efficient solution for stable power delivery.

3.9.4 MP1584

The MP1584 is a compact, highly efficient step-down regulator with an adjustable output voltage range of 0.8V-20V and a maximum output current of 3A. It boasts 90% efficiency and operates at a higher frequency of 500 kHz, allowing for the use of smaller inductors. This makes it particularly useful for low-power circuits where space is limited. In our project, the MP1584 can be used to efficiently regulate power for the HX711 load cell amplifier or touchscreen display, ensuring stable operation while minimizing energy loss.

3.9.5 XL4015

For higher power requirements, the XL4015 is a step-down regulator with an adjustable output voltage range of 1.25V-36V and a maximum output current of 5A. It has an efficiency of up to 96%, making it one of the most efficient regulators for handling high-current loads. With its ability to drive power-intensive components, the XL4015 is well-suited for powering the touchscreen display, ensuring stable voltage delivery without excessive heat generation.

3.9.6 TPS561208DDCR

The TPS561208DDCR is a step-down regulator capable of taking an input voltage of 4.5V-17V & producing an output voltage between 0.76V-7V. It boasts a 580 kHz switching frequency that would allow it to have a fast response time to changes in load, such as when the printer kicks on, the LCD lights up, or the ESP32 transmits via Bluetooth. It has a high efficiency of up to 93%

that is optimized for low power loss, which is important for our project in order to sustain the battery life of our device.

3.9.7 XL6009

The XL6009 is a step-up DC-DC converter that produces an output voltage between 5V-32V. Its ability to boost the input voltage is useful for ensuring that the proper levels are being provided to components in our device that may require different voltages. The regulator has a maximum switching current of 4A and a fixed 400 kHz switching frequency, as well as a high efficiency of up to 94%, ensuring optimum use and performance.

3.9.8 MT3608

The MT3608 is a compact, high-efficiency step-up (boost) switching regulator capable of increasing low input voltages to significantly higher levels, making it well-suited for applications where the power source provides insufficient voltage for downstream components. It can accept input voltages as low as 2V and boost them up to a maximum output of 28V, providing flexibility in systems powered by low-voltage sources such as single-cell lithium-ion batteries.

One of the key advantages of the MT3608 is its efficiency, which typically reaches up to 90% under moderate loads. This efficiency helps conserve battery life and reduces heat dissipation, a critical factor in portable or enclosed systems. The regulator is rated for a maximum output current of 2A, but like most boost converters, this current limit decreases as the output voltage increases.

In our project, while the MT3608 is not suitable for directly powering the ESP32, which operates at 3.3V and is more efficiently supported by a buck (step-down) regulator, it is a valuable option when we need to boost a lower voltage source to power peripherals that require higher voltages. However, for high-current loads such as a thermal label printer, which can require up to 3A or more at 9V, the MT3608's 2A limit becomes a constraint. Attempting to drive such loads may result in voltage drops, overheating, or unstable operation. In these cases, a more robust boost converter is a better alternative. There are options that provide higher current capability and better thermal handling, making them more reliable choices for demanding components in our system.

3.9.9 AMS1117

The AMS1117-3.3V is a linear voltage regulator that provides a fixed 3.3V output from an input voltage ranging from approximately 4.5V to 15V. It is commonly used in embedded systems to power low-voltage digital components, particularly microcontrollers like the ESP32, sensors, and communication modules that operate at 3.3V. Its ease of use and wide availability make it a go-to solution for many development projects.

However, the AMS1117 is a linear regulator, which means it operates by converting the voltage difference between input and output directly into heat. This method of regulation is inherently inefficient, especially when stepping down from higher voltages. Its inefficiency not only wastes energy but can also introduce thermal reliability concerns in compact or battery-powered designs.

The AMS1117 has a maximum output current rating of 1A, which is sufficient for powering an ESP32 under normal operating conditions. However, when additional peripherals such as Wi-Fi modules, displays, or sensors are connected and drawing power, the current demand can exceed this limit, potentially leading to unstable voltage output or regulator shutdown due to overheating or overcurrent.

In our project the AMS1117 can serve as a basic and convenient 3.3V supply for the ESP32. However, for applications requiring better power efficiency, thermal performance, or higher current capacity, a switching regulator is a much better alternative. Unlike linear regulators, switching regulators maintain efficiency across a wider range of input voltages and output currents, significantly reducing heat generation and making them ideal for use in battery-powered or thermally constrained environments.

3.9.10 LM7805

The LM7805 is a widely used linear voltage regulator that provides a fixed 5V output from an input voltage ranging typically from 7V to 25V. It is capable of supplying up to 1.5A of continuous current, making it a straightforward and accessible option for powering moderate 5V loads such as microcontrollers, display modules, or sensors.

Due to its simplicity, the LM7805 is often favored in basic circuits or prototyping environments. It requires minimal external components and provides a solid, noise-free 5V output. However, as a linear regulator, it operates by dissipating excess voltage as heat, which significantly impacts its efficiency. This not only reduces overall energy efficiency but also necessitates the use of heatsinks or thermal management solutions to prevent overheating.

In the context of our project, the LM7805 could be used to power 5V peripherals, such as certain touchscreen displays or sensors, that require clean and stable voltage. However, due to its inefficiency in converting power, especially when stepping down from higher input voltages, it is not the most suitable solution for battery-powered or thermally constrained applications.

3.9.11 LM3478

The LM3478 is a versatile, current-mode boost controller developed by Texas Instruments, designed specifically to drive non-synchronous switching regulators. Its primary function is to efficiently step up lower input voltages to higher output levels, making it an ideal solution for applications requiring compact, high-efficiency power conversion. One of the key advantages of the LM3478 is its wide input voltage range, supporting inputs from 2.97V to 40V, which gives it excellent flexibility in both battery-powered and wall-powered systems. This broad range allows it to operate reliably under varying battery conditions without compromising performance. Additionally, the controller features an adjustable switching frequency up to 1 MHz, enabling designers to optimize for either size (using smaller inductors and capacitors at higher frequencies) or efficiency, depending on the requirements of the application.

The LM3478 utilizes current-mode control architecture, which enhances the transient response and simplifies loop compensation. It also supports features such as soft-start control, shutdown

mode (reducing quiescent current to under 10 μA), and external slope compensation, all of which contribute to stable and efficient operation even under dynamic load conditions. With a peak efficiency of approximately 91%, the LM3478 ensures minimal power loss and heat generation, an important consideration for battery-powered devices like ours, where thermal management and energy conservation are critical. Its compact SOIC-8 package and compatibility with standard external components (inductor, diode, MOSFET) make it well-suited for integration into space-constrained systems while maintaining robust power delivery.

In summary, the LM3478 offers the high-efficiency voltage boosting capability, design flexibility, and reliability required to support the power demands of our printer subsystem, ensuring consistent operation and extending battery life in portable use scenarios.

Table 15: Voltage Regulator Comparison

Voltage Regulator	Type	Output Voltage Range	Max Output Current	Efficiency
LM2576	Buck	3.3V, 5V, 12V (Fixed) or 1.23V - 37V (Adjustable)	3A	~77%-80%
LM2679	Buck	3.3V, 5V, 12V (Fixed) or 1.2V - 37V (Adjustable)	5A	~85%
LM2596	Buck	3.3V, 5V, 12V (Fixed) or 1.23V - 37V (Adjustable)	3A	~80%-85%
MP1584	Buck	0.8V - 20V (Adjustable)	3A	~90%
XL4015	Buck	1.25V - 36V (Adjustable)	5A	~96%
TPS561208DDCR	Buck	0.76V - 7V (Adjustable)	1A	~93%
XL6009	Boost	5V - 32V (Adjustable)	4A	~94%
MT3608	Boost	5V - 28V (Adjustable)	2A	~90%
AMS1117	Linear	3.3V (Fixed)	1A	~50%
LM7805	Linear	5V (Fixed)	1.5A	~50%
LM3478	Boost	9V, 12V, 24V	3A	~91%

3.9.11 Voltage Regulator Selection

We have selected the LM2576 as the buck converter for our project. The LM2576 is highly reliable and easy-to-use. To meet our project's higher-voltage needs, particularly for powering the thermal label printer, we have selected the LM3478 as our boost converter. In our case, it will be configured in a boost topology to step up the voltage from our battery, 3.0V–4.2V, to a stable 9V output capable of supplying the current required by the printer. The LM3478 is known for its wide input voltage range, high efficiency, and adaptability, making it a strong choice for applications where precise control and reliable power delivery are critical.

Together, the LM2576 and LM3478 provide the core power conversion functionality for our system, delivering efficient voltage regulation for both low-voltage digital components and high-power peripherals.

For our final design, we have used the LM2576 to fulfill both our power requirements due to no longer needing a 9V power source. We used it in both our 3.3V and 5V regulators due to its aforementioned reliability.

3.10 Wireless Communication

Wireless communication enables the transfer of information between devices without the need for physical connections, using radio waves or other electromagnetic signals to exchange data over short or long distances. This technology is widely used in modern electronic systems to enhance flexibility, portability, and user convenience.

In our project, wireless communication plays a key role by allowing the seamless transfer of weight data from the scale system to a connected mobile application. Using the built-in Wi-Fi and Bluetooth capabilities of the ESP32 microcontroller, the device can communicate with smartphones or tablets in real time. When a measurement is taken, the ESP32 transmits the weight data wirelessly, enabling users to monitor results directly from their mobile device without needing to interact with the hardware interface.

This wireless connection not only improves user experience but also opens the door to additional features, such as saving historical data, setting dietary preferences, or triggering print commands from the app. It enhances the system's portability and modernizes its functionality, making it more practical for caregivers or healthcare providers who need to track and manage infant feeding sessions on the go. By eliminating the need for wired data connections, wireless communication ensures our device remains compact, user-friendly, and well-suited for both home and clinical environments.

3.10.1 Bluetooth

Bluetooth enables wireless communication between devices using a Bluetooth-enabled microcontroller or a dedicated Bluetooth module. Its low power consumption makes it ideal for our portable design. In our project, Bluetooth would allow the scale to connect to a mobile app for real-time data display. Additionally, it can sync weight readings from the scale to the app, where values could be stored and accessed later.

3.10.2 Wi-Fi

Wi-Fi enables wireless communication over local networks, allowing the scale to transmit weight data to the internet for remote access. Using a Wi-Fi module like the ESP32, our scale could connect to a network and send weight measurements in real time. Since the ESP32 includes both sensor processing and Wi-Fi capabilities, it eliminates the need for an additional module. A major advantage of Wi-Fi is that it allows remote access from anywhere with an internet connection, making it a strong option for cloud-based data logging.

3.10.3 Zigbee

Zigbee is a low-power, low-data-rate communication protocol designed for short-range communication and mesh networking. It is commonly used in applications requiring energy efficiency and reliability. To integrate Zigbee, a Zigbee module can be paired with a microcontroller such as a Raspberry Pi, ESP32, or Arduino. By programming the microcontroller to communicate with other Zigbee devices, weight data can be transmitted to a Zigbee hub. This is particularly useful for applications where data needs to be shared over long distances within a network of connected devices.

3.10.4 Ultra-Wideband

Ultra-Wideband (UWB) is a high-bandwidth, low-power communication technology primarily used for precise location tracking over short to medium ranges. It provides centimeter-level accuracy, making it ideal for positioning systems. UWB can be integrated using specialized UWB modules programmed with a microcontroller. However, since our project does not require precise location tracking, UWB is unlikely to be useful for our scale.

3.10.5 Near Field Communication (NFC)

Near Field Communication (NFC) enables wireless communication between devices within just a few centimeters. To integrate NFC, an NFC module can be paired with a microcontroller, with software programmed to trigger actions when an NFC tag is scanned. The best use case for NFC in our project would be for secure, quick interactions, such as pairing the scale with a mobile app. However, due to its short range and limited data transfer capabilities, NFC is not suitable for continuous data transmission, making it unlikely to be a primary communication method in our design.

3.11 Microcontroller Interfaces

Microcontroller interfaces are essential for enabling communication between a microcontroller and external components such as sensors, displays, and peripheral modules. These interfaces use standardized digital protocols to exchange data and control signals. In our project, the ESP32 microcontroller acts as the central control unit, managing communication with the touchscreen, label printer, status LEDs, and the load cell with its amplifier.

3.11.1 PWM

Pulse Width Modulation (PWM) is a technique used to control power delivery by varying the duty cycle of a digital signal. It is generally used for things like controlling brightness and motor

speed. It can provide backlight support for our touchscreen as well as possible audio feedback and motor control for the printer.

Some advantages include energy-efficient power control, precise control over brightness, and simple implementation. Disadvantages include electromagnetic interference from the constant switching of PWM signals and also power loss from the switching. Other disadvantages include audible noise produced from the switching frequency, possibly dropping to an inaudible range and incompatibility with certain analog systems such as audio amplifiers.

3.11.2 ADC/DAC

Microcontrollers primarily operate in the digital domain, but many sensors work with analog signals where ADC (Analog-to-Digital Converter) and DAC (Digital-to-Analog Converter) come into play. An ADC converts an analog signal into a digital value that the microcontroller can read and process. For example, this would be used for sensor inputs such as the load cell, which would have to convert the outputs received into digital signal. ADCs are also used in battery monitoring by measuring the battery voltage levels to determine the charge status.

A DAC, on the other hand, does the complete opposite and converts digital signal into analog voltage. While DACs are not as commonly needed in basic sensor applications, they are important in areas where analog output is required. Some microcontrollers, including ESP32 variations, come with built-in DAC channels.

3.11.3 Serial Communication Interfaces

Serial communication is essential for microcontrollers to communicate with peripherals and external devices. The most common protocols are UART, SPI, and I2C.

3.11.4 UART (Universal Asynchronous Receiver Transmitter)

UART works by transmitting data one bit at a time in a start-stop manner. Some common applications of UART are for debugging, logging, and communication with modules. UART is often used to send debug messages to a computer and another device. It is also used with many external devices such as Bluetooth modules, which is a huge aspect of our project.

Some advantages of using UART would for one be its simplicity. It only requires two wires, one for transmitting (TX) and one for receiving (RX). Another benefit of UART is that it does not need a clock signal and is already commonly used in many embedded systems.

Some disadvantages of using UART would be that it has a lower data transfer speed compared to SPI and I2C. This would be a minor consideration for us since one of the things we want to minimize in our project is latency and this would be a major contributor to it. Another disadvantage is that each device needs a dedicated UART channel which can make the software composition of our project more complex.

3.11.5 SPI (Serial Peripheral Interface)

SPI is a high-speed, full-duplex communication protocol commonly used for sensors displays and memory chips. Some applications include a display interface which uses SPI for data transfer and flash memory/SD cards which use it for reading and writing external storage. For SPI configuration it uses four main signals MOSI, MISO, SCLK, and SS.

Some advantages of using SPI would be that it is very fast and supports multiple devices using CS (chip select) lines. The disadvantages are that it requires more wires in comparison to UART and I2C and doesn't have built-in error checking. This increases the likelihood of mistakes when configuring our devices and will cause troubleshooting time to be extensive.

3.11.6 I²C (Inter-Integrated Circuit)

I²C is a widely used two-wire communication protocol commonly employed to connect multiple low-speed peripherals to a microcontroller over a shared bus. Some applications of it include sensor communication, commonly used by environmental and biometric sensors, as well as EEPROM storage to read or write to small memory chips, and touchscreen controllers. For I²C configuration, it uses two main lines: SDA (Serial Data Line) for data transmission and SCL (Serial Clock Line) for the synchronization clock.

Some key advantages of using I2C would be that it only requires two wires, devices have their own unique address allowing for easier communication, and it has built-in error detection. The disadvantages are that it is slower than SPI and can be complex when it comes to setting up the addresses.

3.11.7 Two Wire Automotive Interface

Two-wire automotive refers to CAN (Controller Area Network) or LIN (Local Interconnect Network). CAN is a two-wire communication protocol used in vehicles and industrial systems. It allows multiple microcontrollers to communicate without needing a host computer. Common applications are vehicle communication found in engine control units and airbag systems. It is also used for industrial automation in factories and medical devices. Advantages include the fact that it supports multiple devices using a single bus, has robust error detection, and correction. Some disadvantages are that it requires additional hardware for implementation such as a transceiver and is more complex than SPI or I2C.

LIN is a simpler, low-speed alternative for CAN. Common applications include automotive body control where it is used to control things like window motors and seat adjustments. Some advantages of using this type of interface would be that it has a lower overall cost than CAN and requires fewer components. However, it is slower and less robust than CAN.

Table 16: Microcontroller Interface Comparison

Interface	Speed	Wires Needed	Used For	Compatible Components

PWM	Variable	1 per channel	Backlight, motor control feedback audio	LCD backlight, servos, buzzers and DC motors
ADC/DAC	Variable	Analog input/output	Sensor reading, audio output	Load cells, temperature sensors, microphone speakers
UART	Up to 1 Mbps	2 (TX, RX)	Debugging and communication with modules	GPS modules, Bluetooth modules, RFID readers
SPI	Up to 50 Mbps	4 (MOSI, MISO, SCLK, SS)	Display communication and high-speed data transfer	TFT LCDs, SD cards, flash memory and high-speed ADCs
I2C	Up to 3.4 Mbps	2 (SDA, SCL)	Sensors, memory and low-power devices	OLED displays, EEPROMs, IMUs and environmental sensors
CAN	Up 1 Mbps	2	Automotive and industrial automation	Motor controllers, vehicle ECUs, robotics systems
LIN	Up to 20 kbps	1	Automotive body control	Car window regulators, seat controls and lighting systems

3.12 Software Research

In our project, we integrated both a mobile application and embedded software to provide a complete and user-friendly system that supports our hardware. The embedded software running on the ESP32 microcontroller is responsible for managing various subsystems, including data acquisition from the load cell, control of the touchscreen interface, status LED handling, and communication with the label printer. Additionally, this software handles real-time monitoring and decision-making tasks to ensure accurate performance and responsive feedback.

The mobile application complements the embedded system by offering an intuitive platform for remote interaction, configuration, and data visualization. It allows users to view logs of recorded weight readings as well as additional support options. Communication between the embedded system and the mobile app is facilitated via wireless protocols. This integration ensures seamless interplay between hardware and software, enhancing the overall functionality, flexibility, and user experience of the digital scale system.

3.12.1 Mobile Operating Systems

Mobile operating systems are the backbone of mobile devices, managing hardware resources, applications and user interactions. As of today, there are over 6 billion smartphones users and the demand for them is constantly growing. Mobile OS platforms enable developers to create applications that cater to their user's needs. When developing a mobile application selecting the target OS is very critical since it determines the development tools, programming languages, and frameworks that will be used. The two most dominate operating systems today are Apple iOS and Google Android, which collectively make up about 99% of the smartphone OS worldwide. Other platforms such as KaiOS and HarmonyOS serve specific markets but remain secondary to the industry leaders.

3.12.2 Apple iOS

Since its debut in 2007, iOS has redefined mobile technology by offering a seamless and secure experience. Apple has continuously worked to enhance iOS so that it aligns with the evolving user expectations making it one of the most user-friendly operating systems available. iOS is known for its seamless interface that is designed to be accessible even to those with limited technical knowledge. It puts an emphasis on usability, ensuring most functions require minimal setup or customization.

Apple has always placed a strong emphasis on security and data privacy. They have updated iOS to protect user data through many applications including Face/Touch ID, app sandboxing, and strict app store review process. With increasing security concerns in the digital world, it has led many smartphone users to favor Apple devices due to their privacy-first approach.

With that being said, Apple has developed environments tailored for iOS due to their stricter security protocols.

- Swift: A modern, high performance programming language known for its safety and efficiency.
- Xcode: Apple's Integrated Development Environment (IDE) which provides tools for coding, debugging, and testing. Although Apple's strict guidelines provide both quality and security, it can also cause challenges for developers to meet the criteria.

3.12.3 Google Android

Android has been around since 2008 and has since become the world's most widely used mobile operating system. One of Androids most defining features is its open-source nature, allowing developers and manufactures to modify the OS to suit their needs. This type of flexibility enables customization at both the software and hardware level, as well as diverse device options. Even though open-source software enhances flexibility, it also increases security risks. Unlike iOS which enforces a strict app review process, Android does not, leaving it more susceptible to malware and security breaches. Besides its openness, Android also caters to a wide range of price points, making it more accessible worldwide.

Just like Apple, Android provides development environments, offering tools and languages designed for scalability and efficiency.

- Java and Kotlin: Primary coding languages used for Android OS. Kotlin has gained more popularity due to its modern and improved safety features.
- Android Studio: An advanced IDE that simplifies app creation for Android iOS and optimization.

Due to global spread, Android provides developers with more opportunity when it comes to app creation for diverse audiences and device categories.

3.12.4 HarmonyOS

Harmony OS was developed by Huawei and introduced as an alternative to Android for Huawei devices. It is designed for seamless integration across multiple devices including smartphones, IoT gadgets, and smart TVs. Some key features of this OS are that it uses Microkernel-based architecture, is multi-device compatible, and provides support for android applications. While HarmonyOS is still in early development stages compared to Android or iOS it is growing in China and other Asian Markets.

3.12.5 KaiOS

KaiOS is a lightweight mobile OS designed for feature phones. Feature phones are devices that provide essential smartphone capabilities while maintaining a physical keyboard and long battery life. Some key features of this OS include that it supports HTML5-based applications and provides smartphone functions like Google Maps. Although it is a lot less common in smartphones, KaiOS is ideal for some looking for budget-friendly devices in regions with limited internet access.

3.13 Programming Languages

Programming languages are tools that allow us to communicate instructions to computers and microcontrollers. They provide a structured way to define what a system should do, how it should respond to inputs, and when certain actions should be taken. In embedded systems like the ESP32-based digital scale project, programming languages are used to control hardware components, process and interpret sensor data, and create user interfaces.

These languages also allow developers to implement logic and automation, such as detecting when the measured weight exceeds a certain limit and triggering an alert. Different programming languages are better suited to different tasks.

3.13.1 C

C and C++ are fundamental low-level programming languages widely used in embedded systems due to their ability to directly manipulate memory and hardware registers. They follow a procedural programming structure, enabling precise control over system resources and allowing direct access to hardware components. These languages are highly compatible with platforms such as Arduino and ESP32, making them strong candidates for handling real-time sensor data, Bluetooth communication, and printer controls. C/C++ offers efficient execution with minimal abstraction from the hardware, allowing developers to manage memory manually, interact with registers, and integrate with device drivers.

This high level of control and efficiency makes C/C++ a reliable choice for low-power, performance-sensitive applications like our project. However, a notable drawback is the manual memory management and limited built-in support for object-oriented programming, which can increase development complexity and debugging time. While C/C++ require careful handling to avoid issues such as memory leaks or segmentation faults, they remain among the most suitable languages for embedded software development.

3.13.2 Python (MicroPython)

Python is a high-level programming language that supports multiple paradigms, including object-oriented, procedural, and functional programming. It uses dynamic typing, allowing for a beginner-friendly development environment where variable types do not need to be explicitly declared. This flexibility, combined with a vast collection of libraries and frameworks, makes Python highly efficient for a wide range of applications. Its line-by-line execution and availability of pre-built functions enhance code readability and simplify debugging.

MicroPython, a lightweight version of Python, allows developers to run Python code on microcontrollers while offering a faster development cycle compared to C/C++. This is particularly advantageous during the prototyping phase of embedded applications. However, MicroPython generally falls short in terms of performance and stability compared to traditional C/C++ firmware. This is a critical consideration for our project, since python seems best suited for rapid prototyping and this project requires efficient and reliable Bluetooth/Wi-Fi communication and precise control of the printer.

3.13.3 Arduino IDE

Arduino is an open-source platform that simplifies the integration of software and hardware, making it easier to develop embedded systems. The Arduino environment offers extensive support for sensor interfacing and a wide range of communication protocols, including UART, SPI, I2C, Wi-Fi, and Bluetooth. Central to this platform is the Arduino Integrated Development Environment (IDE), which streamlines the process of writing, compiling, and uploading code to microcontrollers. The IDE features a user-friendly interface, built-in libraries, and pre-defined functions, making it accessible even to those new to embedded programming. Additionally, it includes a Serial Monitor for real-time debugging and communication, further enhancing the development experience. With seamless board selection, especially for various ESP32 development kits, the IDE simplifies hardware integration. Overall, the intuitive design and rapid feedback capabilities of the Arduino IDE facilitate efficient prototyping, enabling smooth and iterative firmware development.

3.13.4 Visual Studio Code

Visual Studio Code (VS Code) is an open-source code editor that offers a highly flexible and efficient development environment, making it well-suited for embedded programming. It combines a lightweight, user-friendly interface with advanced features such as intelligent code completion, version control integration, and syntax highlighting. When configured with extensions like PlatformIO or ESP-IDF, VS Code transforms into a fully capable integrated development environment ideal for microcontrollers, supporting serial communication, library

management, and robust debugging tools. Additionally, it accommodates both high-level and low-level programming languages, allowing developers to efficiently write, test, and debug firmware written in C++ or MicroPython. Its versatility and scalability make it particularly valuable for larger, more complex projects that demand greater control, modularity, and long-term maintainability.

3.14 Mobile Application Programming Languages

We utilized programming languages to develop the software that controls and coordinates the various subsystems, such as our microcontroller, mobile application, and user interface on the scale itself. These languages allow us to define the interactions between hardware components and user features.

3.14.1 Java/Kotlin

Java and Kotlin are widely used languages for Android development. Java is a high-level, object-oriented programming language that allows applications to run seamlessly across different operating systems via the Java Virtual Machine (JVM). It promotes modularity, reusability, and maintainability, making it a preferred choice for large-scale projects. Kotlin, a modern statically typed language, requires less code to achieve the same functionality as Java and is often favored for its improved developer productivity and cleaner syntax.

Both languages integrate smoothly with Android's native hardware features, including Bluetooth, local data storage, and system notifications, making them strong candidates for developing an app that communicates with embedded hardware. One major advantage is the ability to fully control performance and ensure compatibility with the microcontroller. However, a key limitation is platform dependency, applications built with Java or Kotlin are Android-only. This means a separate codebase would be required for iOS, leading to increased development time and ongoing maintenance. Despite this, they remain highly reliable choices for Android-focused mobile application development.

3.14.2 Swift

Swift is Apple's modern, general purpose programming language designed for native iOS development. It is recognized for its focus on safety, performance and ease of use. Swift is engineered to be both developer-friendly and highly efficient. It aims to produce software that runs lightning fast, while preventing common programming errors for an optimized performance across apple products. Its concise and readable syntax makes it accessible for beginners, while offering the flexibility to handle complex application development as well. Swift's many safety features include automatic memory management and protection against null pointer dereferencing, which reduces runtime errors. Additionally, it integrates seamlessly with Apple's development tools, making it easy to build responsive and polished interfaces. Swift supports features like Bluetooth, local notifications, and user interface framework to provide an interactive learning environment ideal for learning, experimenting with code and rapid prototyping.

However, similar to Java/Kotlin, Swift is platform-dependent, requiring a separate codebase for cross-platform compatibility. This increases both development effort and long-term maintenance.

Nevertheless, Swift remains a future-focused and efficient choice for applications targeting the Apple ecosystem.

3.14.3 React Native (JavaScript/Typescript)

React Native is a cross-platform mobile development framework developed by Meta that enables apps to be built using JavaScript or TypeScript. It allows developers to maintain a single codebase for both Android and iOS, significantly reducing development time and complexity compared to building separate native apps.

The framework benefits from a large ecosystem, enabling rapid UI development and easy integration with third-party libraries. React Native also supports Bluetooth libraries, which could be used to build the inventory tracking interface and manage user input for this project. However, one key drawback is the performance trade-off due to its abstraction layer. Applications requiring constant Bluetooth communication or intensive data updates may experience reduced efficiency, as the bridge between JavaScript and native code can become a bottleneck. Additionally, debugging may be more challenging compared to fully native solutions. Despite these limitations, React Native remains a popular and effective solution for building cross-platform apps that target both Android and iOS.

3.14.4 Flutter (Dart)

Flutter is a cross-platform mobile development framework developed by Google that uses Dart programming language. Dart is an object-oriented, class-based language with a syntax style similar to C. Flutter enables developers to build high performance renderings applications for both Android and iOS using a single codebase, significantly reducing development time. Flutter's rendering engine allows for near-native performance and offers features like hot reload, where developers can make rapid iterations and test the updated elements together, allowing for an accelerated development process. Additionally, Flutter supports Bluetooth communication, making it well suited for applications that need to interface with embedded devices.

Although Flutter is a cross-platform framework, it delivers performance comparable to native applications. Dart's architecture promotes code scalability, reusability and maintainability. All these qualities make Flutter an optimal choice for a project requiring broader device compatibility without sacrificing performance or user experience.

Table 17: Programming Languages Comparison

Language / Framework	Platform Support	Programming Language	Key Features	Performance
Java/ Kotlin	Android only	Java or Kotlin	Direct access for Android native features, strong support for Bluetooth and hardware communication	High

Swift	iOS only	Swift	Optimized for Apple hardware, modern syntax with high performance and memory safety	High
React Native	Cross-Platform (Android and iOS)	JavaScript/TypeScript	Large plugin ecosystem and hot reload for fast UI iteration on both platforms	Medium
Flutter (Dart)	Cross-Platform (Android and iOS)	Dart	Custom rendering engine for high performance UI on both platforms, near native speed	High

3.15 Code Repositories

Code repositories are centralized storage locations used to store, organize, and manage code and project-related files. They serve as a digital hub where teams or individuals can keep all the files associated with a software or hardware project in one place. Repositories support the tracking of changes made to files over time, allowing developers to monitor progress, revert to earlier versions if necessary, and maintain a clear development history. For our project, using a code repository ensures that our work is organized, changes are documented, and collaboration between team members is efficient and traceable.

3.15.1 GitHub

GitHub is a cloud-based platform that allows developers to create, manage, store, and share code. Built on Git, it provides powerful version control features that support collaboration and track changes throughout the development process. GitHub promotes team-based development by offering both public and private repositories with customizable access permissions. Developers can create branches to experiment or make updates without altering the original codebase, preventing accidental overwrites and enabling organized teamwork. These features make GitHub a strong choice for managing both small projects and large-scale applications while reducing conflicts during development.

3.15.2 Bitbucket

Bitbucket is a Git-based platform designed for collaborative software development. It supports both public and private repositories and includes features such as pull requests, branch permissions, and code reviews to maintain a smooth workflow and high-quality code output. Bitbucket also integrates continuous integration and continuous deployment (CI/CD) pipelines, allowing for automated testing and improved delivery efficiency. Compared to GitHub, Bitbucket has a smaller open-source community, which can make third-party integration more challenging. However, it integrates seamlessly with Atlassian tools such as Jira, making it an excellent choice for teams using modern development and project management practices.

3.16 Databases

A database engine is the core software component responsible for managing the underlying operations of a database. It handles the essential tasks of creating, reading, updating, and deleting data, and governs how information is stored, retrieved, secured, and maintained. The engine ensures that data remains consistent and accessible, even when multiple users or processes are interacting with it simultaneously. Our database engine will be used to track and store logs within our mobile app.

3.16.1 SQLite

SQLite is a lightweight, in-process database engine designed for embedded applications and mobile devices. It does not require a separate server process, making it highly portable and easy to integrate into software. SQLite uses minimal setup, reading and writing data directly to a disk file containing tables, indexes, triggers, and views. However, it is not ideal for high-concurrency applications, as it locks the database during write operations. Additionally, it lacks user-defined functions, limiting its relational capabilities. Despite these drawbacks, SQLite is a suitable choice for lightweight applications where simplicity and low resource consumption are key.

3.16.2 MySQL Database

MySQL is an open-source relational database that is commonly used due to its reliability and performance with large datasets. It supports high-concurrency environments and is able to handle large amounts of data by organizing it into one or more data tables. Developers are able to create, access, store, and modify data with ease making it a popular choice for data management. Nevertheless, with the advantage of scalability this causes more memory to be consumed in comparison to other database structures. Even with this disadvantage, MySQL remains a preferred choice for web development and cloud applications, because of its ease of use, and straightforwardness when it comes to managing and storing data.

3.16.3 Firebase Realtime Database

The Firebase Realtime Database is a cloud hosted NoSQL database, where data is stored in JSON format and synchronized in real-time to connected clients. It ensures data persistence by storing information locally and continues to update even when offline. This gives the user a responsive experience across multiple devices without the need to be connected online. The NoSQL database offers a flexible data structure making it well suited for applications that require real-time updates. However, the firebase has limited capabilities for larger-scale applications causing high data throughput and price. Despite the limitations, the firebase database could be used to sync measurements and timestamps in real-time, making it well suited for our project.

3.16.4 MongoDB

MongoDB is a NoSQL database that can store JSON formatted files making it an ideal database for applications that require dynamic schemas and scalability. It uses horizontal scaling allowing for distributed storage across multiple servers. MongoDB also ensures real-time content management for its clients that require indexing and handling of larger datasets. However, when handling larger datasets this can lead to higher memory consumption and inconsistencies.

MongoDB lacks ACID (Atomicity, Consistency, Isolation, Durability) compliance which ultimately causes instability within the database. Although, these drawbacks are slightly outweighed by the ability to handle dynamic data changes and unstructured data for this project.

3.16.5 PostgreSQL

PostgreSQL is an open-source relational database management system (RDBMS) that supports both structured and semi-structured data. It is fully ACID compliant, offering high reliability, scalability, and extensibility, ideal for complex and data-critical applications. PostgreSQL supports advanced features such as custom data types, procedural languages, and sophisticated indexing strategies, giving developers flexibility in data modeling. The main drawback is its learning curve; optimal performance often requires a more experienced user. Nonetheless, PostgreSQL is a top-tier choice for enterprise-level applications, financial systems, and any use case where data integrity and consistency are paramount.

Table 18: Database Comparisons

Database	Type	Data Structure	RealTime Sync	Offline Access	Advantages	Disadvantages
MySQL Lite DB	Relational (Embedded)	Structured (tables)	No	Yes	Lightweight, fast local storage, low overhead, fully offline	No Realtime sync, not suitable for multi-user
MySQL DB	Relational (Server-based)	Structured (tables)	No	No	Powerful SQL querying, scales well for large datasets	No Realtime sync or offline support, custom backend server needed to support Flutter
Firestore Realtime DB	NoSQL (Cloud-hosted)	JSON	Yes	Yes	Real time updates, offline caching, syncs across all devices	Limited sorting ability for complex filtering
MongoDB	NoSQL	JSON/BSON	With backend	No	Supports unstructured, flexible data, scales well with large data sets	No Realtime sync, backend needed to support Flutter, not fully ACID compliant

PostgreS QL	Relational (Server-based)	Semi-structured	No	No	ACID compliant, supports complex queries, scales well with large data sets	No Realtime syncing without additional setup, requires backend server/ API to support Flutter
-------------	---------------------------	-----------------	----	----	--	---

3.16.6 Database Selection

After researching multiple databases, we have selected the Firebase Realtime Database for our project. While several of the databases discussed above offer robust relational structures, our application required a solution that implements real-time syncing data across devices. The Firebase Realtime Database provides seamless integration with Flutter, along with built-in offline support that automatically synchronizes data upon reconnection. Its ability to deliver live updates ensuring that caregivers and mothers receive timely information, making it the most suitable choice for our database needs.

3.17 3D Modeling Software

For the outer shell of our device, we are considering using 3D printing. This approach is cost-effective, especially since we have access to 3D printers on campus, eliminating the need for external manufacturing services. Additionally, 3D printing allows us to precisely tailor the enclosure to the exact dimensions and layout of our components, ensuring a custom fit that supports our functionality.

3.17.1 Tinkercad

Tinkercad is a free, web-based 3D modeling tool that has an easy-to-use interface. It is great for beginners, educational use and simple designs. Some key features of it include:

- Drag and drop modeling interface
- Large library of pre-made shapes
- Direct export to 3D printing formats (STL, OBJ)

Some advantages of using this software would be the fact that it is beginner friendly and easy to learn how to use. A major disadvantage of using this software would be that it has limited features for complex designs.

3.17.2 Fusion 360 (Autodesk)

Fusion 360 is a powerful, cloud-based 3D CAD software. It offers advanced design, engineering, and simulation features. Key features include:

- Parametric modeling which allows for precise dimensions and assembly
- Cloud collaboration
- Computer-Aided Manufacturing for 3D-printing

The benefits to using this software would be for one that we have prior experience using it in Junior Design. It is free for students and has tutorials for how to do certain designs, but without any prior experience using the software can become very complex and hard to learn how to do.

3.17.3 SolidWorks

SolidWorks is an industry-standard CAD tool used by engineers for creating complex mechanical parts, assemblies, and simulations. It offers advanced features for product design and testing including the following:

- Parametric and direct modeling
- Simulation tools for stress testing and real-world performance
- Assembly design with part management

The advantages of using this software would be that it's good for creating precise, functional parts, has a large library of CAM tools, and we have prior experience utilizing it.

3.17.4 FreeCad

FreeCAD is an open-source parametric 3D CAD modeler. It is primarily used in mechanical engineering and product design but is versatile when it comes to creating technical designs.

Some key features of this software include:

- Parametric modeling
- Has Modularity with a variety of plugins available
- 3D printing integration with built-in export options

This software is free, open-sourced and good for technical designs and projects. However, it can be less user friendly than other CAD software. It also has limited graphic output and animation tools.

3.17.5 OnShape

Onshape is a cloud-based CAD software that allows for real-time collaboration. It's great for team remote work and allows design files to be shared easily. Some features include:

- It is cloud based so no installation required
- Allows real-time collaboration for teams
- Has parametric and direct modeling

The benefits of using this software would be that it is free for non-commercial use and is great for team-collaboration, but it does have limited offline capabilities meaning we could lose our work if a network crash were to occur.

Chapter 4 Standards and Design Constraints

In this section, we will explore the various standards and constraints that influence our project, as well as how we plan to address them in our design process. By identifying these constraints early on we can make informed design decisions that align with industry best practices and project goals. Our approach will involve integrating these requirements into every stage of development, from component selection and circuit design to software implementation and physical design.

4.1 Standards

Within our project, it is essential to adhere to specific standards to ensure that our final product is both safe for users and aligned with consumer expectations. Compliance with established safety, usability, and labeling standards not only improves the overall quality of our design but also increases its reliability and potential for future scalability. During the development and testing

phases, we must take proactive measures to identify and mitigate any risks, particularly when working with electrical components, sensors, and mechanical parts that may pose safety concerns if mishandled.

In addition to safety considerations, we must also ensure that our product's labeling process aligns with official regulations and industry guidelines. This includes accurate and clear labeling of weight measurements, timestamps, and any necessary health or safety warnings on printed output. Adhering to these standards will help establish user trust and ensure the device can be used confidently in everyday settings. Overall, strict compliance with safety and labeling standards will serve as a foundation for delivering a polished, user-centered, and regulation-compliant device.

4.1.1 Electrical and Power Standards

In this section, we aim to address the standards and design constraints that are critical for developing a successful and functional design. These factors include regulatory requirements, material limitations, and technological constraints that will influence our design choices. Following these standards ensures reliability, safety, and efficiency when addressing potential challenges including manufacturability and user experience. All these issues have an impact on the shaping of our final design, which we must keep in mind throughout the development process.

4.1.1.1 PCB Standards

Our PCB layout must adhere to several critical requirements derived from multiple sources, with one key standard being IPC-2221. IPC-2221 is a widely recognized standard developed by the Institute of Printed Circuits (IPC), a global association that establishes guidelines for electronics manufacturing. This standard provides detailed instructions for layout, material selection, signal integrity, thermal management, and manufacturability.

Among the specific design rules outlined by IPC-2221 are those related to trace width, clearance, pad sizes, via dimensions, and layer stackup configuration. Trace width should be calculated based on the current requirements, using formulas specified by IPC-2221A or IPC-2222. For example, for a current of 2A and 1 oz copper, the trace width can be calculated to be approximately 0.336 mm. Clearance between traces depends on the voltage levels, with higher voltages requiring wider spacing to prevent electrical interference. For low-voltage signals, a minimum clearance of 0.2 mm is typically required, whereas for medium-voltage or high-voltage signals, the clearance should increase to 0.3 mm or 0.5 mm, respectively. For power and ground planes, IPC-2221 recommends using copper pours to ensure efficient power distribution and to minimize electromagnetic noise, which is particularly important for ensuring the accuracy of our load cell readings.

To ensure signal integrity, signal traces should avoid sharp bends, instead using rounded corners to reduce reflections. The length of traces should be kept as short as possible to minimize signal degradation and latency. Additionally, trace width should be matched to the impedance of the signal line (e.g., SPI or I2C) to maintain proper signal transmission. For example, SPI or I2C signal traces should be matched to a standard impedance of 50-70 ohms. Given the potential high current draw from components like the label printer, thermal management becomes essential.

IPC-2221 recommends the use of thermal vias and heat sinks to dissipate heat generated by high-power components, such as the printer. Thermal vias should be placed in pads or beneath high-power components, with diameters of 0.5 mm or larger, to efficiently transfer heat to the PCB's back layer for better heat dissipation.

In addition to thermal and signal considerations, the standard specifies the importance of pad sizes and hole sizes to ensure that they conform to standardized dimensions for easier component placement by manufacturers. For example, for components with 0.8 mm lead diameters, the pad diameter should be approximately 1.2 mm, and for through-hole vias, the hole size should be at least 1.0 mm. It is also necessary to ensure that creepage and clearance distances are sufficient to prevent electrical shorts or arcing, particularly in high-voltage areas. For low-voltage circuits, the minimum creepage distance is 0.4 mm, while it should be increased to 1.0 mm or more for higher voltages.

By following these guidelines, our design ensures signal trace widths are optimized for impedance matching with our HX711 load cell amplifier, and the use of power planes reduces voltage drops, ensuring stable operation of the system. The inclusion of thermal vias and copper pours will address heat dissipation, and proper pad and hole sizes ensure manufacturability. Overall, IPC-2221 provides comprehensive guidelines that are crucial for achieving a reliable, manufacturable, and high-performance PCB layout for our project.

As part of our Senior Design course, we must adhere to specific engineering and design criteria throughout the development process. One of the primary requirements is that our final PCB must include at least one microcontroller unit (MCU), a dedicated power supply unit, and multiple peripherals integrated with the MCU. These peripherals must be carefully selected to align with our project's objectives while ensuring full compliance with course expectations. Additionally, our final prototype must not contain any solder-less components, meaning breadboards and jumper wires used during early prototyping must be eliminated in favor of permanent, soldered connections.

Another key requirement is the replacement of development boards with a fully customized PCB. While we may begin prototyping with off-the-shelf modules (e.g., an ESP32 development board), these must be replaced with a custom-designed PCB featuring a well-structured layout. This process involves incorporating the essential circuitry directly onto our board, including voltage regulation, communication interfaces, and peripheral connections, ensuring a professional and manufacturable product.

Furthermore, we are required to integrate existing reference circuit designs into our project. This means leveraging proven industry-standard schematics for key components—such as power management, signal amplification, and communication protocols—and then modifying these designs to suit our specific application. This approach ensures that our final board design is both functional and optimized, reducing potential design flaws and improving overall system performance.

4.1.2 Battery and Power Standards

IEC 62133 is an internationally recognized safety standard issued by the International Electrotechnical Commission (IEC) for rechargeable batteries, specifically those using lithium-ion (Li-ion) and nickel-based chemistries. This standard ensures the safe use, design, and manufacturing of secondary (rechargeable) batteries, which are commonly found in portable electronic devices. For our project, which uses a lithium-ion battery, compliance with IEC 62133 is critical to ensuring that the battery is safe for use by mothers and caregivers. One key aspect of IEC 62133 is the overcharge test, which simulates a scenario where a battery is charged beyond its recommended voltage limit. For lithium-ion batteries, this means exceeding the standard 4.2V per cell. The test ensures that the battery does not overheat, leak, or explode under these conditions. Our device will need to include a Battery Management System (BMS) to prevent overcharging and to limit the voltage to 4.2V per cell, thus ensuring safe battery operation.

Another critical test described in IEC 62133 is the short-circuit test, which evaluates how a battery handles an accidental short circuit. The test ensures that the battery's internal protection mechanisms prevent catastrophic failures such as fires or explosions. To comply with this standard, our design must incorporate protection circuits that detect and mitigate short circuits, ensuring the battery remains safe under malfunctioning conditions. The standard also outlines the importance of battery durability through an impact and drop test, which simulates physical accidents. For our project, this is crucial as the portable scale may be handled frequently and subject to drops or impacts. The test ensures that the battery casing can withstand physical stress without compromising the internal battery structure. To meet these requirements, we must use robust battery casings that can endure typical wear and tear.

Additionally, the standard specifies a thermal test to ensure that the battery operates safely at high temperatures. The battery must be able to handle up to 130°C without leakage or failure. Since our scale may involve high-power components such as the label printer, it's essential to ensure that the battery can handle the heat generated by these components. Incorporating thermal management solutions, such as heat sinks or thermal vias in the PCB layout, will be crucial in ensuring the lithium-ion battery remains within safe operating temperatures. The standard also stresses the importance of limiting charging current and voltage, recommending a 1C (1mA per hour) charging rate for lithium-ion batteries. This ensures that the battery charges at a safe rate that prevents excessive heat buildup during charging. Our charging circuitry will need to be designed to handle less than 1C charging rates, preventing overheating and ensuring the battery remains efficient over time.

Overall, adhering to IEC 62133 ensures that the lithium-ion battery in our portable breastmilk scale is safe to use. By following these rigorous testing protocols and design recommendations, we can prevent issues such as thermal runaway, short-circuiting, and physical damage to the battery, ensuring that the scale remains safe and reliable for the user. This includes integrating a Battery Management System (BMS), ensuring robust casing for impact resistance, and incorporating proper thermal management features. These measures are essential not only for user safety but also for regulatory compliance and product reliability.

4.1.3 LED Standards

As a part of our advanced goals, we will be implementing a visual indicator using LEDs. Due to this there are a variety of standards we need to conform to ensure the safety and efficiency of our device.

4.1.3.1 IEC 62031

This standard focuses on the safety requirements for LED modules used in general lightening applications. It provides detailed criteria to ensure that LEDs are safe to use in various environments and applications. The main aspects covered by this standard are the different types of standards such as electrical safety, electromagnetic compatibility, requirements for marketing, durability/lifespan and more.

As it relates to independent LED modules built into the hardware of a product, the standards lay out a commitment to transparency of manufacturing by requiring products to disclose the mark of origin (trademark, manufacturer's name or name of the responsible vendor/supplier) as well as the model number or type reference of the manufacturer. For us, this would mean listing the specific models of LEDs we use, which is listed here in Chapter 10 section 10.2. The component transparency extends to the range of component values; for example, the safe operating voltage of the LED needs to be stated to the user to ensure proper usage of the device. For our selected LEDs, the operating voltage is between 3.2-3.5V. By following these standards, we can ensure that the LEDs used in our project function properly and are not a safety hazard to other users.

4.1.4 Wireless Communication Standards

The device will use Bluetooth Low Energy (BLE) for connectivity to the app so it must comply with the IEEE 802.15.1 standard for Bluetooth and the Bluetooth SIG specifications for BLE. Compliance with these standards will ensure secure and efficient data transfer as well as allow compatibility with multiple devices.

IEEE 802.15.1 are described under the Wireless Personal area network. Bluetooth SIG established the specifications for BLE (Bluetooth Low Energy) which is when IEEE added a standard for it. BLE must operate in the 2.4 GHz ISM band using Gaussian Frequency Shift Keying (GFSK) modulation which ensures minimal power consumption and reliable data transfer. BLE follows a layered protocol stack which defines how the data is transmitted and received. At the physical layer BLE supports multiple modes including 1Mbps, 2Mbps, and LE Coded PHY which allows for optimal data rates and a range of up to 100 m depending on the environment. The link layer manages device connections, security and packet transmission. The Generic Attribute Profile (GATT) and Attribute Protocol (ATT) define how structured data is exchanged between connected devices. Additional layers such as SMP and GAP provide encryption, authentication and device role management.

Our breastmilk scale functions as a peripheral that advertises its presence and waits for a central device (ie. smartphone) to establish a connection. This setup ensures energy-efficient communication since the scale remains in a low-power state when it is not actively transmitting data. We will also be implementing the previously mentioned BLE mode to reduce the power drain when the device is idle.

4.1.5 Breastmilk Storage Standards

There are several important guidelines for the safe storage and handling of breast milk, as recommended by the CDC. Breast milk should be stored in breastmilk-specific storage bags or clean, food-grade glass or plastic containers with tight-fitting lids. Each container must be clearly labeled with the date the milk was produced, and if the milk is intended for childcare or donation, the child's name should also be included. Proper storage location is essential. Milk should be kept at the back of the refrigerator or freezer instead of the door to avoid temperature changes. To reduce waste, it is best to freeze milk in small portions of two to four ounces. There should be at least one inch of space left at the top of each container to allow for expansion during freezing. When traveling, expressed milk can be stored safely for up to twenty-four hours in an insulated cooler bag with ice packs. Regarding storage time, freshly pumped milk can sit at room temperature for up to four hours, refrigerated milk is good for up to four days, and frozen milk is best used within six months but can be kept for up to twelve months. Thawed milk should be used within twenty-four hours and should never be refrozen. The oldest milk should always be used first following the first in, first out principle.

These guidelines have helped shape the design and features of our breastmilk management system. Our project combines a digital scale, touchscreen display, and label printer to help mothers store and track expressed milk more easily. By including these recommendations in our system's software, users will receive printed labels with the date, volume, and optional dietary restrictions. The device will also send reminders when milk is approaching expiration to help prevent waste and ensure safety. The mobile app will log all milk entries and guide the user to use the oldest milk first. By turning public health recommendations into helpful tools and features, our system supports mothers in managing breastmilk safely and effectively while reducing daily stress and decision making.

4.1.6 Security and Software Protocols

Our project includes a mobile application that will use Bluetooth Low Energy (BLE) to transmit data between the device and the app. Given the sensitivity of the data being exchanged, it is essential to implement robust security protocols to protect user information and ensure the integrity of our system.

4.1.6.1 ISO/IEC 27001 (Information Security Management)

ISO/IEC 27001 is an international standard for Information Security Management Systems (ISMS). It outlines the requirements for establishing, implementing, maintaining, and continually improving an organization's information security management system. For our project, adhering to ISO/IEC 27001 is crucial because the device and mobile application will be collecting and transmitting sensitive personal health data, such as milk storage records and other health-related information. By following this standard, we can ensure that the data is securely handled and protected from unauthorized access, which is paramount to safeguarding user privacy and ensuring compliance with relevant regulations like HIPAA (Health Insurance Portability and Accountability Act).

One of the key aspects of ISO/IEC 27001 is its focus on risk management, which involves identifying potential security threats and vulnerabilities and mitigating them through effective

controls. Specifically, the standard advocates for end-to-end encryption to secure data at all stages of its transmission. For example, as our device will be using Bluetooth Low Energy to transmit data between the scale and the mobile application, the data must be encrypted to protect it from unauthorized interception. BLE itself supports pairing and bonding mechanisms, which can be used to establish secure communication between the device and the app, ensuring that only authorized devices can connect and exchange data.

Moreover, ISO/IEC 27001 specifies the implementation of audit trails and logging mechanisms, which allow the tracking of access to sensitive data and any changes made. This is critical in detecting potential security breaches or unauthorized activities. By incorporating this feature into our system, we can provide a transparent record of all interactions with the device and the data, which could be crucial for both security audits and regulatory compliance.

Furthermore, ISO/IEC 27001 mandates the regular review and improvement of security practices to adapt to emerging threats and vulnerabilities. As our device will continuously evolve, ensuring that security measures are regularly updated and tested will be essential to maintaining a high level of data protection.

By adhering to the guidelines set forth in ISO/IEC 27001, we can ensure that both the device and mobile application handle personal health data securely, protecting user privacy and ensuring that sensitive information is transmitted and stored in compliance with the highest standards of data security.

4.1.6.2 ISTQB Software Testing Standard

The International Software Testing Qualifications Board (ISTQB) is an internationally recognized organization that provides a structured approach to software testing, ensuring the reliability, performance, and quality of software systems. ISTQB's testing methodologies and frameworks are designed to cover all aspects of software development, from functional testing to security and performance testing. For our project, which includes firmware development, touchscreen UI design, mobile application functionality, and wireless communication integration with the microcontroller, adhering to ISTQB guidelines will be crucial to guarantee that each component operates as expected and that potential issues are identified early in the development process.

One of the core principles of ISTQB is the V-Model of software testing, which emphasizes the importance of verifying and validating software at each stage of development. This model involves planning tests early in the software development lifecycle and performing them in parallel with the development process. For instance, when developing the firmware for our ESP32 microcontroller, we will apply unit testing to ensure individual functions and modules work as expected. This can include tests for sensor readings, data processing, and communication with peripherals such as the HX711 load cell amplifier.

In addition to unit testing, integration testing will be crucial for validating the interactions between the mobile app, firmware, touchscreen UI, and wireless communication components. Given that our device will transmit weight data via Bluetooth Low Energy, we must ensure that data transmission between the device and the mobile app is secure, reliable, and error-free.

System testing will verify that the entire system works cohesively, from the touchscreen UI capturing user inputs to the printer receiving and printing the weight data accurately.

Regression testing is another important aspect of ISTQB's methodology that will apply throughout our project. As we integrate new features or fix bugs, regression testing ensures that existing functionality is not broken by these changes. This will be particularly important when adding features such as secure BLE communication or when enhancing the mobile app with additional functionalities for managing breast milk storage and health data.

Performance testing will also be necessary to ensure the device responds quickly and operates efficiently, particularly when handling real-time sensor data and managing print jobs. Given that our device involves data handling, authentication, and communication over BLE, it's crucial that the system can operate within the expected time frame under typical usage conditions. Load testing may also be required if our mobile app is expected to handle multiple users simultaneously or large amounts of data.

ISTQB also places a strong emphasis on security testing to ensure that sensitive data is protected throughout the software's lifecycle. In our case, security testing will involve verifying that personal health data, such as breast milk storage and usage records, are securely transmitted over Bluetooth Low Energy, properly encrypted, and stored with secure authentication mechanisms. By following ISTQB security testing guidelines, we will mitigate risks associated with unauthorized access or data breaches, which is particularly important given the sensitive nature of the information.

Overall, adopting ISTQB guidelines throughout the software testing process will ensure that the firmware, mobile app, touchscreen UI, and wireless communication components are thoroughly tested, free from critical bugs, and meet the performance and security standards required for a reliable and trustworthy device. By doing so, we will improve the quality of our product, identify issues early in development, and ensure the final system meets the needs of users, particularly in terms of usability, security, and reliability.

4.2 Constraints

There are several critical constraints that we must consider throughout the development of our project, each of which has the potential to significantly impact our workflow, design decisions, and overall progress. These constraints include factors such as time limitations, budget restrictions, scheduling, and testing challenges related to integrating our components. As a team, it is essential that we remain aware of these constraints at every stage of development and use them to guide our decision-making process. By proactively addressing these limitations, we can ensure a more efficient workflow, avoid costly setbacks, and ultimately deliver a successful and well-functioning final product.

4.2.1 Testing Constraints

Before integrating any components into the final system, it is essential to test each part individually to verify proper functionality and identify any potential issues early in the development process. Isolated testing helps us confidently determine whether a malfunction is

due to the component itself or a flaw in our system design. However, there are several constraints we must carefully manage during this phase. Due to us having a limited timeline for the semester every stage of development must be meticulously scheduled and followed to avoid falling behind.

Unexpected delays, such as defective parts, late shipments or prolonged troubleshooting can immensely push the timeline off track, reducing time for thorough testing and correction. For example, one of the most sensitive components is the load cell, which is prone to damage if subjected to weights beyond its rated capacity. As such, we must take extra precautions when conducting weight tests, especially when working near the upper limit of the load cell's range. This includes using calibrated test weights and closely monitoring readings to avoid overstressing the sensor. In our case, some components, like the load cell amplifier or microcontroller, may have long lead times or limited availability which could stall the integration phase if replacements are needed.

Additionally, because the project is being built and tested in a shared lab environment, access to tools, workbenches, or power supplies may be restricted based on availability. The Senior Design lab is used by multiple teams, all working towards similar deadlines, which can create a competitive overcrowded workspace. This can limit the amount of testing we are able to complete during any given week. As the semester progresses and the project deadline approaches, the demand for lab space and equipment will likely increase. This can result in scheduling conflicts, reduced bench time, and limited access to high-demand equipment like soldering stations. With challenges such as these, it could delay not only hardware debugging but also software testing and integration. Furthermore, some equipment must be signed out or reserved in advance such as the Hantek oscilloscope and the availability of equipment such as this determines how much testing can be done outside of the labs.

To mitigate these constraints our team has created a structured development timeline that clearly defines milestones, prioritizing early testing of critical components and keeping a buffer period for unexpected revisions. Due to the fact that we are working within the confines of our Senior Design course we must also ensure our project progress aligns with expected deadlines. This often means that design decisions must be finalized earlier than ideal, limiting our ability to explore alternate approaches or incorporate feedback. As a result, opportunities for repeated trials are reduced meaning we must be more efficient with each design cycle.

To balance this demand, we have decided to divide our project into subsystems to be tested and corrected independently even as other components are progressing. This modular approach will allow us to identify problems quickly and maintain steady progress across different workstreams. We also strive to perform as much preliminary work as possible outside of the lab such as code development and breadboard prototyping to limit the need for extended lab time during busy times. In addition, we have implemented a documentation system to record testing outcomes and revisions. This improves not only collaboration among the team but also ensures the traceability of changes.

Ultimately the success of the project depends not only on the design itself but also on our adaptability when troubleshooting with limited time. By maintaining flexibility within a well-

defined plan and emphasizing early testing, we aim to meet the goals and objectives of our project without compromising quality or functionality.

4.2.2 Time Constraints

One of the most significant constraints for our project is time. We have a total of 16 weeks in Senior Design I and only 9 weeks in Senior Design II due to the condensed summer semester. This compressed schedule requires careful planning and efficient execution to ensure we meet all project requirements within the allotted time. In comparison to teams who have the traditional full two semesters, we must be more strategic with how we manage overlap between design, implementation, and testing. Every decision made in the early weeks will greatly impact on our ability to succeed in the final month of development.

Our hardware design process will commence in March 2025, and we must have a fully functional prototype completed by early August 2025. As such, we must approach the timeline with careful planning and disciplined execution. We must prioritize key tasks such as component selection, procurement, hardware integration, software development, and testing to ensure timely completion. Since each phase builds on the progress of the previous one, a delay in any task can cause a setback for the entire timeline.

Each of these tasks are heavily interdependent. For example, calibrating the scale and testing the weight-sensing algorithm relies on a functioning load cell. The load cell must be successfully integrated with a calibrated amplifier and a compatible microcontroller. This means delays in ordering parts, waiting on shipping, or struggling with hardware compatibility could push necessary development work weeks behind schedule. As Senior Design II progresses, some parts may be deemed unusable if they cause too many issues for projects throughout the course. This can be a time setback in itself due to schematics having to be redone, which means reworking the hardware and software that are already done. This makes time management critical throughout each step of the process. We must allocate sufficient time for component selection, ordering, and delivery, especially since hardware delays are common occurrences that may affect integration. This makes early sourcing components a priority, especially since a common issue seen in Senior Design is the delay in PCB design and ordering phase. In our case, core components like microcontrollers, load cells, and printers can vary due to supply chain issues or limited stock. To mitigate this, we plan to identify and order long-lead time components as early as possible to avoid bottlenecks later in the development process.

In parallel, software development must progress steadily to allow time for debugging, testing, and refinement. Mobile app development, Bluetooth integration, and Real-time data synchronization, cannot be completed in isolation. These features are heavily dependent on the functional communication between hardware and software. For instance, without a working Bluetooth module that communicates effectively with the microcontroller, progress on the app's data interface cannot continue. Unexpected issues with Bluetooth or Wi-Fi communication, sensor calibration, or printer reliability could arise, making these integration challenges become unpredictable and time-consuming. These integration challenges often tend to appear once multiple components are connected, making them very time-consuming and unpredictable to resolve.

Another time constraint is the academic schedule itself. Multiple deliverables, including design reviews, peer evaluations, and report deadlines, can split our focus and pull time away from actual development or testing. Balancing documentations requirements with physical prototyping is critical to take into consideration for our plan. Additionally, the summer schedule means less time between classes and fewer breaks to catch up, making the project feel more rushed. If major setbacks occur later in the project timeline, we may have limited opportunities to implement corrections, which increases the importance of proactive planning and execution during the early weeks.

Not only do we have to worry about allocating time for design, implementation and testing but also for research. This is our first time building something fully on our own without an instruction manual given to us, so we must utilize research. Research is a big part in how we plan our implementation and execute it, especially when it comes to things like the load cell. We have never used one before and have no prior experience of how to utilize it but by doing research we are able to grasp a concept of where to start. Learning about these components does take a lot of time, especially considering we have to do it at every stage. At the design phase we must research components and past products to determine how and what we are going to use for our project. At the implementation stage we have to research how best to connect our components. At the debugging/testing stage using research to help figure out possible issues with our project. With that being said this only makes it harder to allocate time to certain areas because it may be needed elsewhere.

To mitigate these risks, we will establish clear milestones, track progress regularly, and address potential bottlenecks proactively. We plan to use tools such as shared Gantt charts, weekly team check-ins, and version control systems to stay up to date as well as identify problems early on. Regular meetings between the team and project sponsors will ensure accountability and provide opportunities to adjust accordingly if setbacks are encountered. Additionally, assigning periods of accommodation for hardware debugging and late-stage testing will ensure the ability to make changes instead of cutting corners to meet the end goal. We will also document our testing results, bugs fixes, and integration steps to a shared log to ensure we don't duplicate or lose track of changes made.

Another strategy we will implement is concurrent workflow execution wherever possible. While one team member finalizes hardware integration, another can continue refining the user interface or updating documentation. By dividing our project into subsystems, such as power regulation, weight sensing, or app communication, we can test features independently before final integration. This not only minimizes risk but also gives us flexibility in adjusting individual subsystems without hindering the rest of the project.

By adhering to a well-structured timeline and maintaining flexibility for unforeseen challenges, we aim to deliver a robust and fully operational prototype within the given timeframe. Our ability to manage time effectively will be essential to meet both project requirements and ensure our final project is reliable, safe, and usable.

4.2.3 Ethical and Social Constraints

Since our project is sponsored, the intellectual property (IP) rights for the final design and its associated software legally belong to our sponsors. As such, we are both ethically and legally obligated to ensure our design complies with relevant IP regulations and does not infringe on any existing patents. This requires comprehensive research into existing technologies and patent filings to avoid potential legal conflicts. Given that scales with integrated label printing already exist, it is our responsibility to ensure that our final design introduces unique features or innovations that differentiate it. This may involve modifying functionalities, improving efficiency, or integrating novel elements that do not overlap with patented technologies. It is our duty as engineers to respect the innovation of others while striving to create meaningful contributions of our own.

In addition to IP concerns, we also carry ethical responsibilities related to user data privacy and security. Our project involves tracking breastmilk production, storage, and usage for individual mothers, which are all sensitive information that must be handled with care. When this data is shared with caregivers or integrated with milk banks, strong data management practices are essential. This includes secure storage, encryption, and user consent protocols to prevent unauthorized access. As the system evolves, we must continuously evaluate and implement best practices for ethical data handling. For instance, if integrated with hospital systems, the device must comply with HIPAA (Health Insurance Portability and Accountability Act) standards. This ensures that all collected health data remains confidential and accessible only to those with proper permissions.

From a social perspective, our project is intended to support mothers, infants, and caregivers, which places a heightened responsibility on us to ensure the system is safe, reliable, and accessible. A simple and intuitive user interface is crucial to accommodate users who may be experiencing fatigue, stress, or limited technical experience. This means minimizing cognitive load, technical language involved and ensuring a smooth user experience. We must also consider physical accessibility, such as button size, screen brightness and ease of interaction for users multitasking. With potential commercialization in mind, designing for accessibility and ease of use is a priority to ensure equitable access for all users.

Furthermore, we must also consider inclusivity in our design by creating a system that is not only helpful to the “typical” user but for users with different physical and cognitive abilities. Ethical engineering also involves questioning implicit assumptions and striving for solutions to accommodate diverse user scenarios. For instance, can the interface be understood by users with visual impairments? How will language or cultural differences impact how data is entered or interpreted? These are all things that should be taken into account to ensure our device meets all users’ needs.

In terms of testing, we must also be cautious in how we simulate user interaction and handle any data. Ethical testing involves informed consent, anonymity of data and sensitivity to the users’ emotional or physical well-being. For example, our user base includes postpartum mothers, who may be in physically and emotionally delicate states. We have to avoid burdening users and approach each phase with care and compassion in mind.

Finally, we have a broader ethical obligation to approach our design with empathy. Our sponsors, who bring both personal and professional insights as parents and engineers, play an active role in this process. Their involvement allows us to remain grounded in real-world needs and better understand the experiences of mothers who pump and store breastmilk. Their feedback will be a critical part in helping us align our technical goals with real-life expectations. In this way, the design process becomes a collaboration between the developers and end users. By recognizing and actively addressing these ethical and social constraints, we hope to produce a final product that is both technically reliable and socially responsible.

4.2.4 Economic Constraints

Our sponsors have generously allocated a working budget of \$500 to support the development of our prototype. This amount is expected to cover most of our expenses, including essential components and materials. Fortunately, our design is relatively simple in terms of mechanical complexity, with only a few moving parts. However, a significant portion of our budget is dedicated to the thermal printer, which appears to be the most expensive component based on our research and cost estimates. The printer is critical to the core functionality of our system, as it is responsible for producing the labels that track breastmilk storage data.

Given the printer's importance to our system, we may need to make trade-offs between cost and performance, especially if more affordable printers come with technical limitations or challenges. Other necessary expenses, such as the microcontroller, load cell sensors, power supply, PCB printing, and prototyping materials, will further tighten the budget, leaving little room for error or reordering.

As much as we'd like to stay within our sponsors' allotted budget, we may need to contribute our own personal funds to bring this design to life. Unforeseen circumstances may require higher-specification components or replacements that the current budget does not account for. Our goal is to deliver a reliable and fully functional prototype that meets all our project requirements. To achieve this, we must make strategic financial decisions and manage component usage carefully, keeping in mind that some out-of-pocket investment may be necessary.

4.2.5 Manufacturability Constraints

For the hardware aspect of our project, all the components need to be housed in an enclosure that is compact, portable, and easy to use. Although no one in our group has prior experience with any 3D printing design programs, it may be our best option in terms of price and speed to create the enclosure, as opposed to laser-cutting or outsourcing the manufacturing process to a third-party. Additionally, our purchased printer will already have its own enclosure that will need to be considered in our design since we would have to be able to access the label paper in order to refill it. The printer's charging method would also need to be considered in our overall design's power supply & PCB design.

Something worth noting is that from a course perspective, our enclosure and design could be basic enough to the point where considerations such as the aforementioned printer paper tray and charging port could simply be accessed through holes or cutouts in the enclosure. However, this project was conceptualized by our sponsors who want a design that can be manufactured commercially in the future. With that in mind, the design we came up with for this project needs

to reflect a higher standard beyond simple functionality. Whether or not our sponsors reuse our design in its entirety, our design needs to be sleek and marketable enough to hopefully alleviate or dispel the need to be redesigned in the future.

4.2.6 Realistic Design Constraints

Another important consideration for our project is the set of practical design constraints that affect functionality, performance, security, and scalability. One of the most significant technical challenges lies in hardware limitations that directly impact software development. For instance, microcontrollers typically offer limited processing power and memory, which restricts the complexity of embedded software. As a result, the software must be carefully optimized for efficient memory usage to prevent crashes and ensure reliable operation. Features such as Bluetooth communication, real-time data processing, and UI responsiveness must be balanced against these limitations to achieve the best possible performance. Additionally, many microcontrollers have minimal onboard storage, requiring data to be offloaded to an external database. This introduces further constraints related to memory management and the reliability of wireless communication. Altogether, these factors demand careful RAM optimization and robust data handling to maintain system stability.

Portability is another key constraint. To keep the system lightweight and battery-operated, the hardware must be energy efficient. However, features like continuous Bluetooth or Wi-Fi communication can quickly drain the battery. To mitigate this, the software must minimize transmission times and implement low-power protocols whenever possible. While these strategies may impose certain limitations on the user experience, they are essential for maximizing the system's operational longevity and achieving our portability goals.

Chapter 5 Comparison of ChatGPT with other Similar Platforms

The concept of artificial intelligence is a broad one that has many applications in the modern day. In 1950, Alan Turing proposed the idea that by the turn of the century, computer-generated communication would be indistinguishable from human communication, at least 70% of the time (Codecademy). In the last 70 years, the research into AI has been developed through various specific points of interests, whether it be medicine (MYCIN, 1972) or music (WABOT, 1980's); but most importantly in 1965 with the chatbot ELIZA, one of the first AI programs to imitate human conversation. It is this branch of AI use that has led to the development of chatbots such as ChatGPT, Google Gemini, and Microsoft Copilot; all of which are largely used as search engines for various information, and which most directly relates to our usage of AI in modern education.

What we understand as AI is achieved by high-speed processing that allows machines to make decisions based on access to existing data. The process begins with machine learning, where computers use statistical models and algorithms to facilitate its own learning without needing to be taught by human input. The goal is to imitate the self-developing learning capabilities of humans to continuously improve on its knowledge base and thus its effectiveness in decision-making; a process called deep learning. As its understanding and knowledge expands, it compartmentalizes related information into neural networks (named as such because of its imitation of the networks of neurons in the human brain), where it improves its predictive

capabilities by extrapolating on patterns within closely related nodes of information. Alongside these processes of learning development, AI also utilizes natural language processing which allows it to understand and mimic human speech in order to communicate data with human users as well as accept inputs and search prompts. In addition, AI regularly uses computer vision, where it pulls from the aforementioned learning algorithms to process and analyze an image to learn, as opposed to writing and words.

Research has always been a difficult process without any aid, seeing as most people have used some form of online search engine for every inquiry ranging from personal to professional. AI chatbots & search engines have recently seen a surge in usage, likely due to its ability to condense research from various sources and respond to inquiries with more insight than regular search engines, all while mimicking the conversational tone of other human beings. However, this useful and innovative tool is not without its drawbacks and repercussions, which we will be discussing in this section as well as its impact on education and this project, specifically.

5.1 Platform Comparison

There are a multitude of AI platforms being marketed today, each boasting different capabilities and strengths. In this section, we will be examining three different AI platforms and comparing them to each other with their pros and cons: ChatGPT, Google Bard (Gemini), and Microsoft CoPilot.

5.1.1 ChatGPT

ChatGPT is a generative AI chatbot developed by OpenAI. It uses a generative pre-trained transformer (GPT) model, meaning it pulls from textual data that is fed to it and is able to produce human-like responses. It uses natural language processing to engage users in conversations while being able to search its wide database for responses to queries. ChatGPT is largely credited for the rise in AI platforms, as most platforms (including the ones mentioned later) debuted after ChatGPT was released in 2022. Primarily existing to emulate human conversation, the platform is very versatile and is able to respond in a variety of media types. Its database is fairly limited, though, as its most recent knowledge cutoff is December 2023 (according to the platform itself). There is also a paid version of the platform that offers more features including lasting memory of your conversations, faster optimized performance, and the ability to input different file types.

5.1.2 Google Bard (Gemini)

As is the case with many of the platforms following the release of ChatGPT, Gemini offers many of the same capabilities that ChatGPT does, albeit with different emphases in different aspects of its response. For example, Gemini keeps memory of your conversations in the basic free version, while ChatGPT reserves that feature behind a paywall. Gemini also has native integration with Google, not only giving its knowledge database more frequent updates, but also allowing it to access information on your Google Drive and Google Docs accounts - ideal for those using AI as a daily companion. Where Gemini pales in comparison to ChatGPT is with its responses, which are both slower to produce and not as robust, limiting the different media types it is able to respond with.

5.1.3 Microsoft CoPilot

According to the platform itself, Microsoft Copilot aims to provide personal responses that are meant to strike a deeper emotional resonance with the user. Once again, many of the features it provides overlap with ChatGPT (i.e. multimedia response, web access database, paid pro versions, etc.), though ChatGPT provides faster responses with more powerful processing. Copilot offers the ability to take more time to develop its response to better tailor itself to your needs, but the tradeoff is a roughly 30 second additional process time from the basic version. Similar to Gemini, Copilot has native integration with Microsoft developed programs such as Bing.

5.2 Benefits

For most of its users, AI is generally interpreted as a net positive for how useful it is in assisting with everyday needs and even more complex tasks during research, making it a very popular tool amongst students. In this section, we will discuss three aspects of AI that are beneficial to users: specificity, inspiration, and multimedia responses.

5.2.1 Specificity

While most search engines will suffice with finding generally available information or answering simple factual questions, many times during research a question will arise that cannot be answered as easily with a single prompt. AI platforms have the advantage here; whereas most search engines are simply compilations of available sources and articles that are then sorted through for relevance to your inquiry, AI can take several relevant sources and use them to develop a solution to a more complex question. In addition, AI can actively improve its response by asking more about the prompted inquiry, rectifying gaps in its own knowledge and understanding in an attempt to better suit the needs of the user.

5.2.2 Inspiration

In addition to the specificity, AI can be a useful tool in guiding research. Specifically with instances like Senior Design where the end goal is known but the work leading up to it is up for interpretation, AI and its expertise can offer a place to start. For those who may be lost on what exactly to ask/search, AI can accept a general set of instructions/directions as opposed to a specific question and suggest next steps to follow - something general search engines like Google cannot do. This assistance is true for any stage in the design process; in whatever step a person may be stuck on. Simply explaining the problem at hand to AI can lead to a breakthrough and facilitate progress throughout the entire project.

5.2.3 Multimedia

Generative AI is not only useful as a search engine. Many AI platforms including ChatGPT are capable of providing more than just text-based responses, they can also create a document for various applications. AI can generate schematics, mockups, diagrams, data tables, scripts, codes, and many more resources depending on the needs of the user.

5.3 Disadvantages

While AI is generally very useful, the benefits tend to overshadow some of the more harmful repercussions to both the environment and its users. In this section, we will discuss three disadvantages of AI: environmental concerns, inaccuracy, and academic misuse.

5.3.1 Environmental Concerns

Most AI platforms are run by data centers that, according to the UN Environment Programme (UNEP), use a large number of resources to keep running. UNEP cites an estimate of water usage that outnumbers that of the entire country of Denmark 6 times over, as well as hazardous electronic waste (such as mercury and lead) as a byproduct. The construction of the computers dedicated to running the AI platforms also requires 800kg of raw materials for a 2kg unit. Finally, UNEP states that a single ChatGPT prompt consumes 10 times the electricity of a Google Search.

5.3.2 Inaccuracy

The tradeoff for instantaneous responses can sometimes be factual errors. Depending on the platform in use, the information being relayed may contain outdated or otherwise inaccurate information, if the platform's database is not updated regularly. In addition, ChatGPT does not always cite its sources, inhibiting users from fact checking the response for themselves. This would lead to users having to do their own research through other search engines and come to their own conclusions; invalidating the use of the AI platform in the first place.

5.3.3 Academic Misuse

As previously stated, Generative AI is capable of producing many different types of responses. Academically, students may feel inclined to take advantage of this and use AI to generate papers, essays, or codes for assignments that are meant to assess the students own skills and knowledge. Although the source is different, this misuse can be classified as plagiarism and can lead to violations of university codes and ethics if discovered, or a lack of the aforementioned skills and knowledge if not.

5.4 Examining AI Usage for Our Project

For our project, AI provided a helpful tool for gathering information on various components, as well as hardware and software guidance. Below, we will examine some of the prompts we asked our AI platform of choice (ChatGPT) and discuss how the responses we received impacted our project overall.

5.4.1 Case Study 1

In our research for the hardware of the project, we needed to consider how to supply power to our device. After researching different methods of power supplies, we decided that our best choice was a lithium-ion battery (section 3.3.12). From there, we then had to decide what specific lithium-ion battery component we wanted to use; which is where we decided to consult ChatGPT for assistance. The prompt we asked was **“what would be the best lithium-ion batteries to use for the project”**. The response is documented in Appendix E.1.

ChatGPT provided four options, each with six bullet points highlighting what the component was best used for, battery capacity, voltage, size, rechargeability, and a quick support for why that component should be used in our design. While the structure of the response was concise and easily understandable, two out of the four options were not what we had asked for. One of these two options was for a battery pack that differed from the specific battery cells that we had in mind. The second extraneous option that ChatGPT provided us with was a lithium-polymer battery, which ran into the same problem as the previous option where it is not a type of lithium-ion battery and therefore does not contribute to our project.

Both cases highlight how AI can sometimes misinterpret what is being asked and can instead suggest alternatives seemingly relevant to the topic, but which do not actually apply. As both of these options were presented as fact with the real lithium-ion options, it seems that ChatGPT does not understand that these are not all the same, which demonstrates how AI can still have a gap in knowledge/understanding that can hinder the reliability and accuracy of its responses.

On the other hand, we did end up using the other two options provided in our design considerations, and one of them was ultimately the specific component we ended up choosing for our project. While ChatGPT had its issues, the response was still able to provide useful information that helped us make important decisions.

5.4.2 Case Study 2

For Chapter 4 of this report, we were required to examine different standards and constraints that our project would have to adhere to or abide by. On top of all the research required for building and designing our project, this chapter required us to consider the real-world expectations of our project as a product in a greater commercial and socioeconomic context. To expedite the research of this field, we utilized ChatGPT to assist us in finding out what specific standards and constraints would factor into a product like ours. To do this, we first fed ChatGPT a prompt explaining the objectives of our project as stated earlier in the report in Chapter 2 so that it would be able to conceptualize the product we intended to create. After that, we asked it the prompt: **“What are some standards and constraints that should be considered for this product?”**. The response is documented in Appendix E.2.

Similarly to the last case, the response from ChatGPT had varying degrees of utility and relevance to our project. From a constraint perspective, there weren't many insightful suggestions that we hadn't considered before, seeing as the constraints overall are mostly ones relating to our team specifically: what we are or aren't able to do with the resources and circumstances we have. Despite this, the response from ChatGPT introduced us to many electrical and software standards that we hadn't known about prior such as ISO/IEC standards on quality and the UL/IEC standards on safety.

The extraneous content in this case study highlights a general oversight that AI tends to have. For this case, ChatGPT recommended health and safety standards “for food-contact materials” by the FDA and the DNV (ISO). Upon further research, these standards only apply to food products being marketed and sold and therefore are not relevant to our project. The oversight in question lies in AI assuming food to have a larger role in our project, presumably stemming from

the mention of breastmilk. As stated before, with the batteries, AI can occasionally misinterpret the intent of the prompt/user and resort to providing information that is only tangentially related to specific key words or phrases in the prompt.

5.4.3 Case Study 3

For our hardware design, we are required to provide schematics for the different components. Since the AI was able to take the summary of our project from Chapter 2 and gain an understanding of all the different parts required, we prompted ChatGPT **“Help me find reference schematics”** and it was able to provide a compiled list of websites and sources for each component/system in our design: Load Cell with HX711 Amplifier, Microcontroller (ESP32) Integration, Thermal Print Integration, and Power Management. Each section provided between 1-3 sources that ranged from YouTube videos, data sheets, and other projects and blog posts discussing the schematics and connections between our components.

As mentioned in section 5.1.1, AI can be very specific in understanding a prompt in a way that other search engines would not, pulling from the sustained knowledge it is able to acquire from conversation between the user and the AI. It is then able to expedite the process of research by applying a single prompt to several fields of relevance simultaneously and providing more curated information than a simpler search engine could. Beyond the schematics asked for in the prompt, ChatGPT was also able to provide tutorials of project connections and sample codes for software.

The only critique with this response is that the Power Management section only provided general documentation on the ESP32 instead of sources relating to the battery module or voltage regulators, as the header would imply. Overall, this assistance was able to impact the ease of research for the project by extrapolating on its knowledge of our needs.

Chapter 6 Hardware Design

Our hardware design comprises several core components, each serving a vital function to ensure the overall system operates reliably and efficiently. This chapter presents the individual circuit schematics that we have consolidated into a printed circuit boards (PCB). Each subsystem is described in terms of its functionality, wiring configuration, and role within the broader system architecture.

6.1 Breadboard Demonstration

As an initial validation step, we prototyped our circuit on a breadboard to confirm that all components functioned correctly before committing to a finalized PCB layout. A working demonstration video of this breadboard prototype was submitted alongside our final report.

For this phase, the key hardware elements include an ESP32 microcontroller, an HX711 load cell amplifier, a load cell sensor, and a 3.5” ILI9488 SPI touchscreen display. The load cell was securely mounted between two plates, forming a stable platform for accurate weight measurement. It was interfaced with the HX711 module, which amplified the millivolt signal from the load cell and transmitted a digital output to the ESP32. The ESP32 processed this data

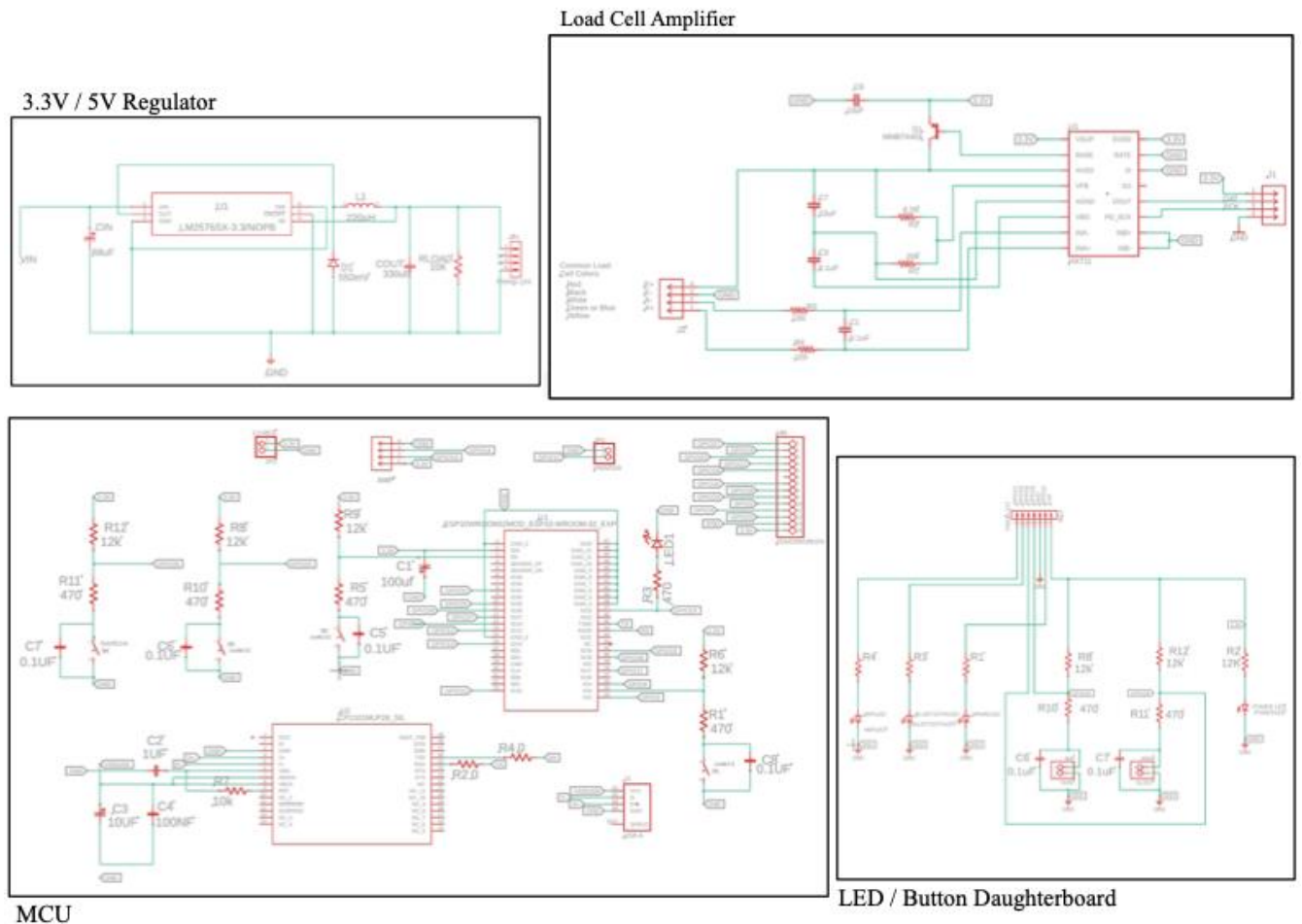
and displayed the weight in real time on the touchscreen interface. We used the Arduino framework to upload our test code and calibrated the load cell using the HX711 library, ensuring accurate and stable readings.

This breadboard prototype allowed for testing and iteration of the sensor integration, touchscreen display output, and Bluetooth connectivity prior to final hardware integration.

6.2 Overall Schematic

We have combined all our schematics into a singular image so we can have an overview of our significant PCB design. This integration is essential for the development of a cohesive and space-efficient PCB design that supports all required hardware functionalities. The centralized schematic not only streamlines the routing process but also ensures consistency in signal integrity and power distribution across the system. In the final design, each of these components has been integrated onto a custom PCB daughter board tailored to our device's form factor, power requirements, and portability constraints.

Figure 5: Overall Schematic

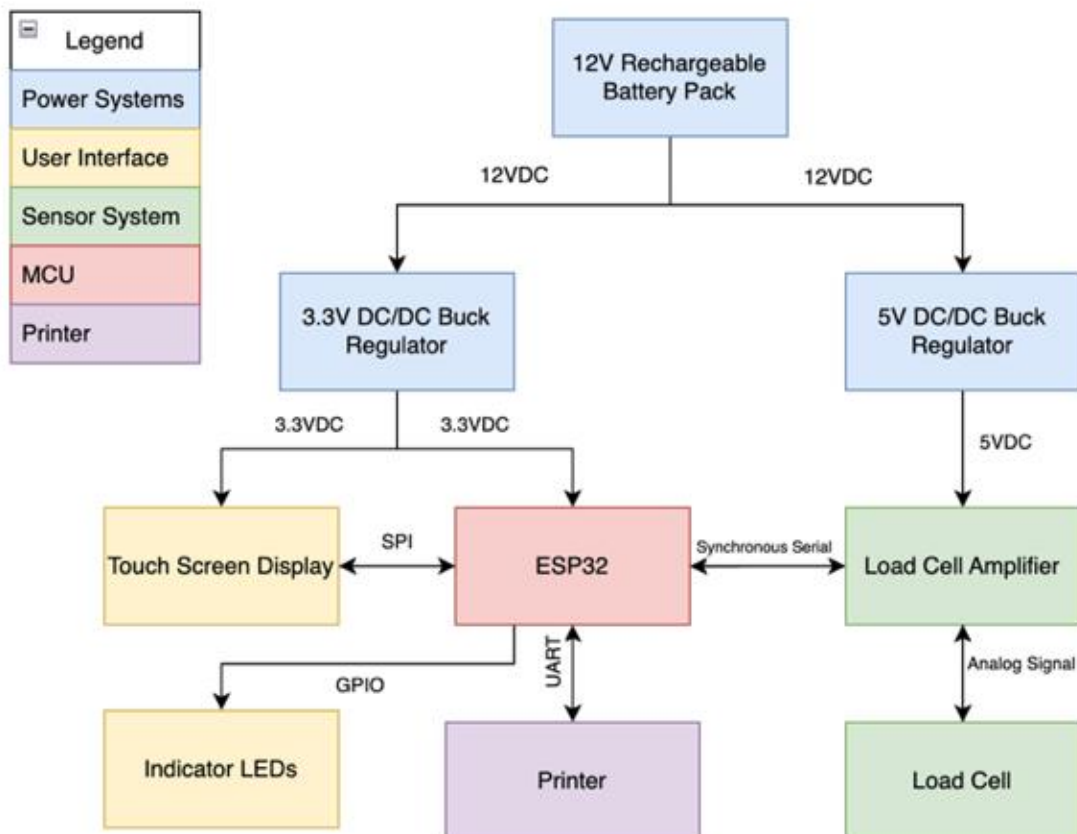


6.3 Subsystem Block Diagram

This block diagram illustrates the primary subsystems of our digital scale with an integrated label printer. At the heart of the system is the ESP32-WROOM-32 microcontroller, which functions as the central control unit. It manages all communication and coordination between the connected peripherals, including the load cell amplifier, the touchscreen display, and the label printer. The microcontroller processes weight data from the load cell, displays relevant information to the user via the touchscreen, and sends print commands to the printer as needed. Power for the device is primarily supplied through our portable battery.

From the battery, two voltage regulators distribute power to the system components. We directly use 12V from our battery to supply power to the label printer, which requires a higher operating voltage to function properly. A 3.3V buck regulator powers the other subsystems, including the ESP32 microcontroller, and the touchscreen display. The HX711 load cell amplifier module is powered through a 5V regulator. This separation of power ensures that each component receives a stable and appropriate voltage level.

Figure 6: Subsystem Block Diagram



6.4 Converter Schematics

Our design incorporates various components and subsystems, each with distinct power requirements. To ensure proper voltage and current levels, we have implemented buck DC/DC converters.

6.4.1 9V DC/DC Boost Converter Schematic

To supply a stable 9V power rail for the thermal label printer, a DC/DC boost converter architecture is implemented using the Texas Instruments LM3478 high-efficiency boost controller. The primary function of this converter is to elevate the voltage from the lower output range of a single-cell lithium-ion battery, typically 3.0 V to 4.2 V, to a constant and regulated 9.0 V output. This regulated output is capable of sourcing up to 3A of continuous current, meeting the power demands of the thermal print head, particularly during peak load conditions such as active printing cycles where both voltage stability and high current delivery are critical.

The LM3478 operates as a current-mode, non-synchronous boost controller, which provides precise control over output regulation and allows the selection of external MOSFETs and diodes to optimize performance for the specific power requirements of the load. Its wide input voltage range (2.97 V to 40 V) makes it highly suitable for battery-powered applications, while its internal error amplifier and oscillator enable designers to configure the converter for both light-load efficiency and high-load capability.

The converter circuit includes a high-frequency switching inductor, an external N-channel MOSFET, and a fast recovery Schottky diode, chosen for their efficiency and thermal performance characteristics. This component selection and layout ensure that the circuit can sustain efficient operation under dynamic load conditions without inducing thermal stress on critical components.

To minimize voltage ripple and support high transient response, low equivalent series resistance ceramic capacitors are placed at both the input and output stages. The output voltage is sensed through a resistive feedback divider connected to the internal error amplifier of the LM3478, ensuring tight regulation and system stability across varying load conditions. This power stage achieves a peak efficiency of approximately 91%, which is crucial in extending battery life and reducing thermal dissipation, especially in compact enclosures with limited passive cooling. The compact physical footprint of approximately 769 mm², including all supporting passive components, allows for efficient integration into our design.

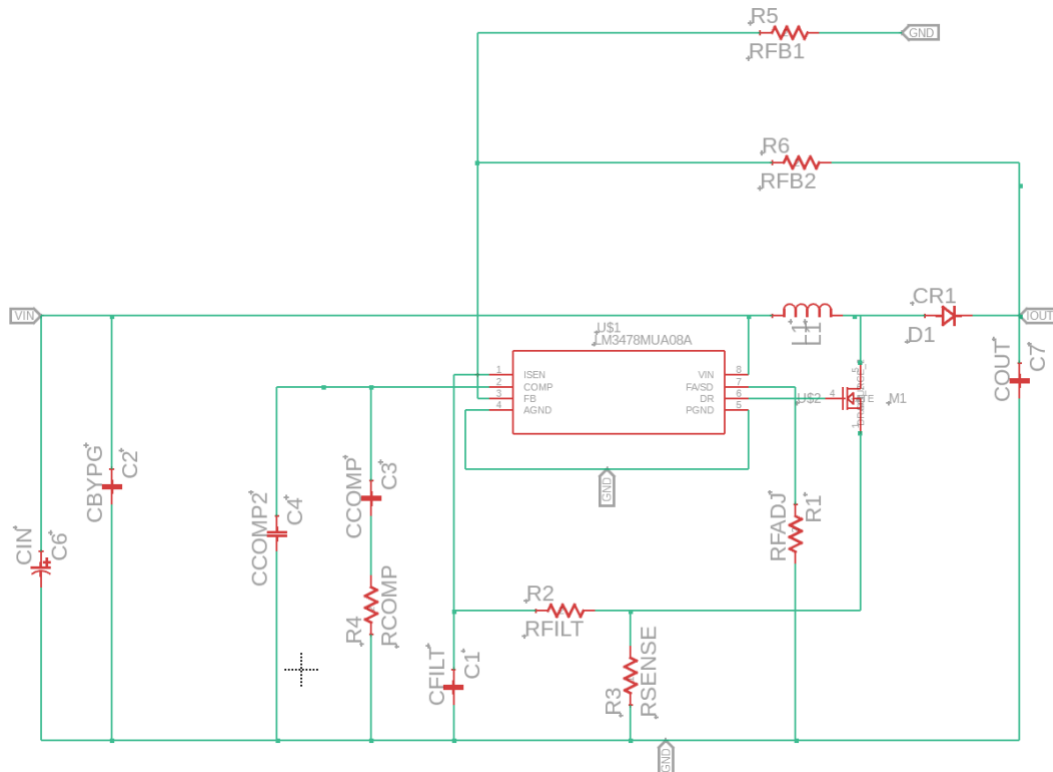
Furthermore, the estimated bill of materials (BOM) cost is \$3.76, which offers a balance between performance, footprint, and cost. This cost-efficiency is achieved without compromising output reliability or thermal stability, making the LM3478-based converter a practical solution for powering energy-intensive subsystems such as thermal printers.

In summary, the LM3478-based boost converter represents a critical component of the system's power delivery architecture, enabling dependable and efficient 9V output from a single-cell battery source. Its high efficiency, configurable design flexibility, and compact size make it

ideally suited for modern portable applications where space, power, and thermal constraints must be simultaneously addressed.

For our final implementation, our power requirements did not require the use of a boost regulator for 9V, as we supplied power directly from our battery pack. This schematic was unused in our final design.

Figure 7: 9V DC/DC Boost Converter Schematic



6.4.2 3.3V/5V DC/DC Buck Converter Schematic

To supply a stable 3.3V power rail for the system's digital and analog subsystems, a DC/DC buck converter based on the LM2576SX-3.3 voltage regulator has been used. This regulator is a fixed-output step-down switching regulator that simplifies the power supply design process by eliminating the need for an external resistor divider network, thereby reducing design complexity and minimizing the overall bill of materials. The fixed 3.3V output is particularly suitable for microcontroller-based systems and peripheral components that require tightly regulated voltage levels.

The converter is designed to operate with an input voltage ranging from 8V to 16V, sourced from the output of the system's battery charger. To suppress high-frequency noise and input ripple, a 68 μ F electrolytic capacitor (CIN) is placed across the input terminals. This filtering

capacitor ensures a stable input voltage and protects the switching regulator from voltage transients.

During operation, the LM2576SX-3.3 modulates current through a 220 μH inductor (L1), which stores energy and smooths current flow to the output. A Schottky diode (D1) with a forward voltage drop of 550 mV is placed in the path to allow current flow when the internal switching element is turned off. The choice of a Schottky diode enhances overall efficiency due to its low forward voltage and fast switching characteristics.

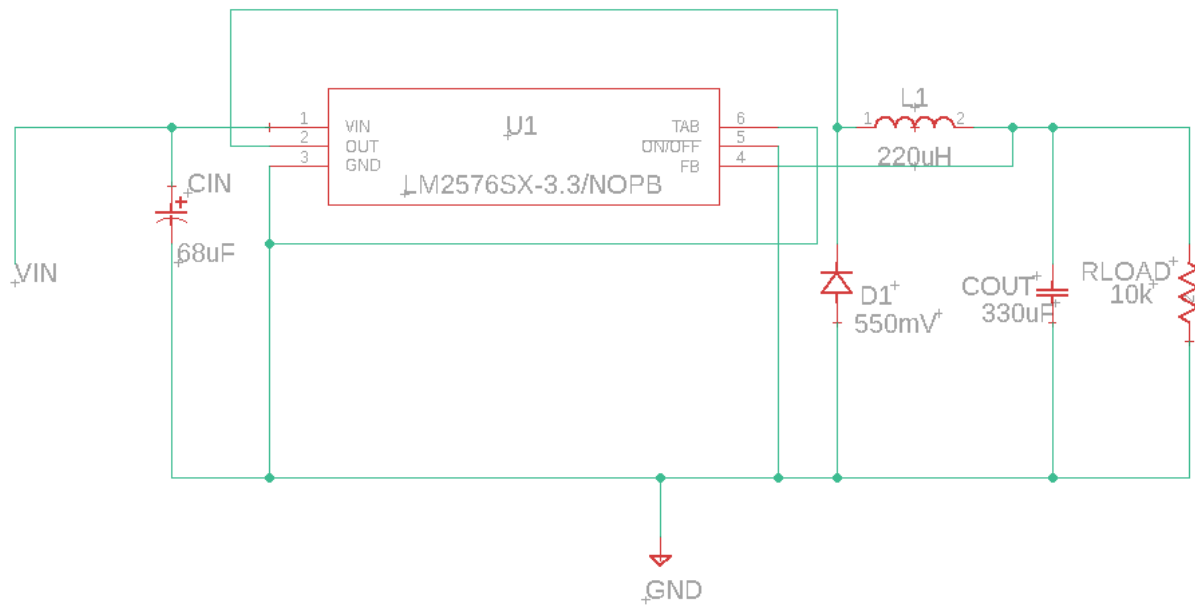
On the output side, voltage stability is maintained using a 330 μF output capacitor (COUT), which helps to reduce output voltage ripple and ensures a smooth DC output under dynamic load conditions. A 10 k Ω resistive load (RLOAD) is connected across the output to simulate the typical current draw of the system and to maintain a minimum load for stable converter operation.

The LM2576SX-3.3 operates at a fixed switching frequency of 52 kHz, which enables a balance between component size and power conversion efficiency. The regulator also incorporates internal current limiting, thermal shutdown, and output short-circuit protection, providing a high level of safety and fault tolerance. The circuit is capable of delivering up to 1A of continuous output current at 3.3V, with a measured conversion efficiency of 83.6% under nominal load conditions.

This configuration results in a compact implementation, with a total PCB footprint of 915 mm², and a component cost of \$8.74. Overall, the LM2576SX-3.3-based converter offers a cost-effective, efficient, and reliable power management solution well-suited for embedded system applications where stable 3.3V power is essential.

To supply a stable 5V rail required by several components in our system, we modified our original power regulation approach. Instead of redesigning the entire schematic, we retained the same layout and components but replaced the fixed-voltage version of the regulator with an adjustable (ADJ) version of the LM2576 switching regulator. This allowed us to fine-tune the output to 5V by selecting appropriate resistor values in the feedback network, while maintaining the efficiency and simplicity of our original design.

Figure 8: DC/DC Buck Converter Schematic



6.4.3 Battery Charging Module

To manage battery charging and power path control in our system, we utilize the MCP73837, a highly integrated single-cell Li-ion/LiPo charge management controller. This device not only ensures safe and efficient charging of our 18650 lithium-ion battery but also facilitates seamless switching between external power sources and battery power. The MCP73837 accepts dual input sources, either a regulated wall adapter (VAC) or a USB port (VUSB), allowing the system to operate flexibly depending on available power. When both sources are present, the controller prioritizes the wall adapter for charging, while still allowing the USB to serve as a secondary input. Input capacitors, denoted as CIN1 and CIN2 in the schematic, are included to suppress voltage ripple and noise, ensuring clean input power for the charging circuit.

Once powered, the MCP73837 manages the charging process using a well-defined algorithm that includes preconditioning, constant-current, and constant-voltage phases, followed by charge termination based on current thresholds. The positive terminal of the battery connects to the VBAT pin, while a parallel output capacitor (COUT) provides voltage smoothing and helps manage transient loads. One of the key features of the MCP73837 is its integrated power path management, which allows the system to operate directly from the input power when available and automatically switches to battery power when the input is removed. This transition is facilitated by an external P-channel MOSFET (Q1) and a Schottky diode (D1), which together ensure continuous power delivery to the system without requiring manual intervention or additional control logic. The RPULL resistor ensures proper biasing of the gate during this transition.

To provide user feedback and charging diagnostics, the STAT1, STAT2, and PG output pins drive status LEDs through current-limiting resistors (RLED1, RLED2, and RLED3). These indicators inform the user of the charging state, completion status, and power source availability. Furthermore, the MCP73837 includes a thermistor interface (THERM), which connects to an

directional conversion makes it possible to upload firmware, monitor serial output, and send runtime commands from a development environment like the Arduino IDE, PlatformIO, or Espressif's own ESP-IDF.

Beyond simple data transmission, the CP2102-based bridge also supports additional functionality that enhances the development process. It can supply regulated power (either 3.3V or 5V, depending on configuration) to the ESP32 and other connected peripherals, simplifying power management during prototyping. The module includes onboard LED indicators that display transmission (TX) and reception (RX) activity in real-time, offering visual feedback that is helpful for diagnosing communication issues or confirming successful data transfer.

Physically, the USB side of the bridge connects to a standard USB Type-A or micro-USB connector that plugs into the computer. The UART side typically provides a breakout for TXD (transmit), RXD (receive), GND, and optionally RTS and CTS for hardware flow control, though only TX, RX, and GND are essential for basic serial communication. These UART signals interface directly with the ESP32's serial pins—usually GPIO1 (TX) and GPIO3 (RX) for programming, allowing the host system to flash firmware images onto the microcontroller and interact with the serial monitor for debugging and data logging.

Furthermore, the CP2102 chip supports USB 2.0 full-speed operation (12 Mbps), and is recognized across major operating systems, including Windows, macOS, and Linux, often without the need for manual driver installation. Its robust support for baud rates up to 1 Mbps ensures that it can keep pace with high-speed serial applications, making it ideal not only for programming but also for runtime data streaming applications such as sensor data acquisition or logging system diagnostics.

In conclusion, the USB to UART bridge is an indispensable tool in embedded system development, acting as both a programming interface and a live debugging channel. Its integration into our ESP32-based platform enables efficient firmware deployment, real-time serial communication, and simplified power delivery, all of which are vital to the streamlined development and testing of our embedded application. The choice of the CP2102 ensures compatibility, reliability, and ease of integration, making it a practical solution for both prototyping and eventual deployment.

Figure 10: USB to UART MCU schematic

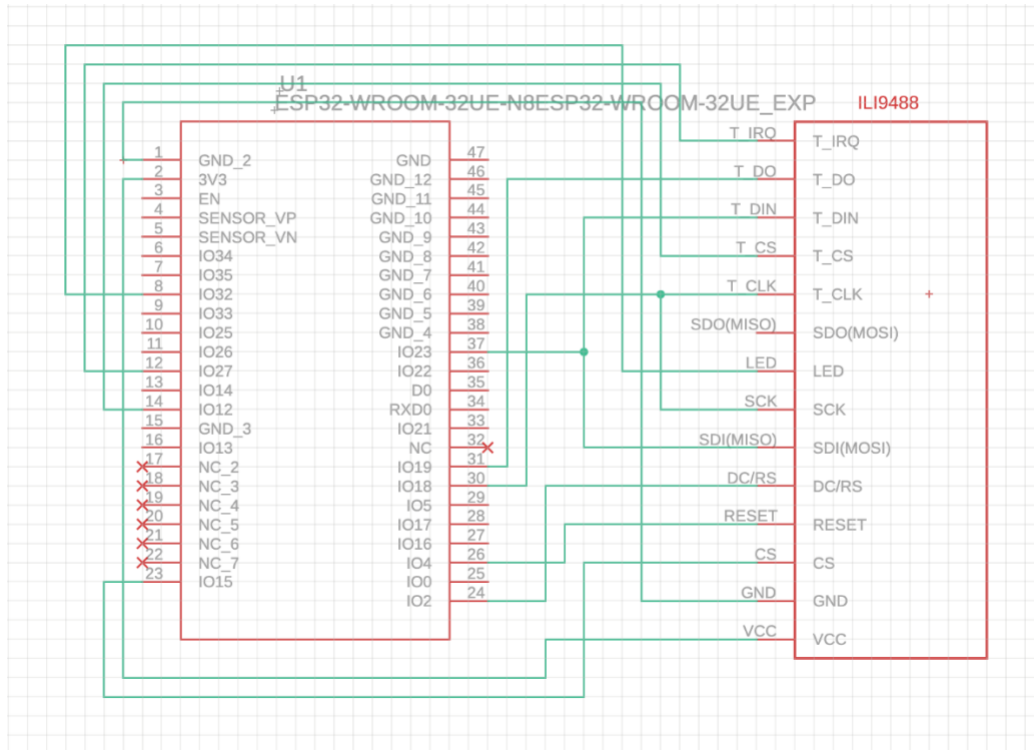
6.6 Touchscreen Schematic

In addition to display functionality, the ILI9488 also integrates a resistive touchscreen, which operates separately within the SPI framework. The touchscreen functionality is enabled by

connecting the T_IRQ pin (GPIO 33) for detecting touch events, T_DO (GPIO 25) for transmitting touch data, T_DIN (GPIO 23) for receiving data from the ESP32, T_CLK (GPIO 27) for synchronizing touch data, and T_CS (GPIO 14) for selecting the touchscreen controller. These connections allow the ESP32 to accurately interpret touch inputs while simultaneously rendering graphical content.

By properly wiring these connections, the ESP32 controls the multiplexed rows and columns within the TFT panel, activating specific pixels to render images. The integrated touchscreen controller provides an interactive interface, making it possible for users to interact with the display. With its high-speed SPI communication and efficient control signals, this interface ensures a responsive graphical display with touch capabilities, which will be essential for our portable digital scale user interface.

Figure 11: Touchscreen to MCU schematic



6.7 Weight Sensor Schematics

Our design incorporates load cells as weight sensors for the scale. These load cells, which detect applied weight, have been interfaced with a load cell amplifier to ensure accurate signal processing. The load cell amplifier amplifies the weak signal output from the load cells, allowing the microcontroller to accurately measure and interpret the weight data. This amplified signal is then sent to the microcontroller, which will process the data for display and further use in the scale's functionality.

6.7.1 Load Cell Amplifier Schematic

This schematic illustrates the load cell signal conditioning and amplification circuit, which forms a critical part of the system's weight measurement subsystem. At the core of this circuit is the HX711, a 24-bit analog-to-digital converter (ADC) specifically designed for weigh scale and industrial control applications. The circuit serves as an interface between the load cell sensor and the ESP32-WROOM-32 microcontroller, enabling precise measurement of small voltage variations produced by mechanical force on the load cell.

The load cell, positioned on the left side of the schematic, typically consists of a Wheatstone bridge configuration with four strain gauges. It has four primary terminals: Excitation+ (E+), Excitation- (E-), Signal+ (S+), and Signal- (S-). These wires connect directly to the HX711's input pins, where E+ and E- provide the necessary excitation voltage to power the bridge, and S+ and S- deliver the resulting differential voltage signal generated by deformation of the load cell. The output voltage from the load cell is typically in the millivolt range and must be amplified before digitization.

The HX711 features two differential input channels, Channel A and Channel B, which are accessible via the pins labeled INA+, INA- (for Channel A) and INB+, INB- (for Channel B). In most weigh scale applications, only Channel A is used because it offers a programmable gain amplifier (PGA) with gains of 128 or 64, ideal for amplifying the weak signals from the load cell. AVDD provides the analog supply voltage to the internal analog circuitry, while VBG is the internal reference voltage that ensures stable operation and accurate conversion results. The highly integrated nature of the HX711 eliminates the need for external amplification or additional signal conditioning circuitry, significantly simplifying the design.

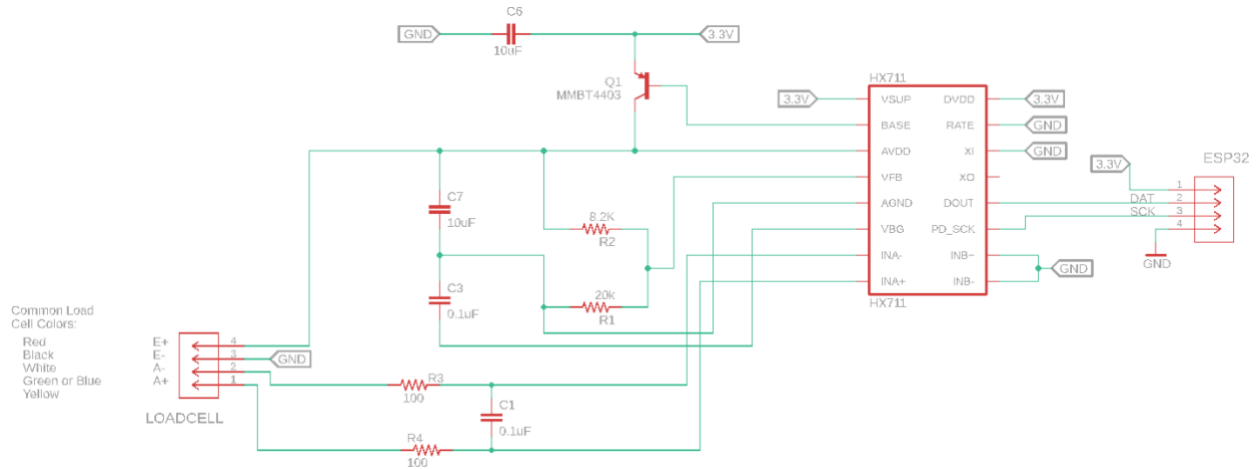
Data communication between the HX711 and the microcontroller is facilitated by two digital lines: PD_SCK (Power Down and Serial Clock Input) and DOUT (Data Output). These pins implement a synchronous serial communication protocol similar to SPI, though not fully SPI-compliant. The microcontroller generates a series of clock pulses on PD_SCK to read 24 bits of digital data from DOUT, which corresponds to the amplified and digitized weight signal. This minimalist interface allows for efficient data acquisition with minimal overhead and is well-suited for low-power embedded systems.

To interface the amplifier module with the ESP32, a 5-pin male header is provided on the right side of the schematic. The pins include VCC (3.3V), VDD (5V), LD_CELL (serial data line), SCL (serial clock line), and GND (ground). The dual power supply pins, VCC and VDD, offer flexibility in powering both the analog and digital sections of the HX711, depending on system voltage levels. In our implementation, VCC (3.3V) is used to match the logic levels of the ESP32, while VDD (5V) may be used to supply the load cell excitation if required. The serial data line (LD_CELL) and serial clock line (SCL) correspond to the DOUT and PD_SCK pins of the HX711, respectively, and are connected to designated GPIOs on the ESP32 to handle digital communication.

By converting the minute analog output of the load cell into a high-resolution digital signal, the HX711 enables accurate and consistent weight readings in real-time. Its integration into the system not only enhances measurement precision but also simplifies hardware complexity due to

its built-in PGA, reference voltage, and serial interface. This configuration provides a compact, efficient, and cost-effective solution for weight measurement, forming a foundational component in our portable digital scale design.

Figure 12: Load Cell Amplifier Schematic



In our final implementation, we chose to base our custom load cell amplifier design on the reference schematic provided in the HX711 datasheet, rather than using the SparkFun breakout board schematic. This decision allowed us to integrate only the essential components required for our application, optimize the layout for our system's needs, and reduce dependency on prebuilt modules.

6.7.2 Load Cell Schematic

The load cell in our system is configured in a Wheatstone bridge arrangement, a standard and highly effective method for precisely measuring small changes in resistance that occur in response to mechanical strain. This configuration consists of four strain gauges arranged in a balanced bridge network, where two gauges typically experience tensile strain while the other two experience compressive strain. When no load is applied, the bridge remains balanced, and the differential voltage output is close to zero. Upon the application of force, such as weight being placed on the platform, the resistance of the strain gauges changes proportionally to the strain experienced, resulting in a measurable voltage difference across the bridge output terminals. This small differential voltage, typically in the range of a few millivolts, serves as the analog signal representing the weight applied to the load cell.

To make this small analog signal usable for digital processing, it must be accurately amplified and converted into a digital format. This is accomplished using the HX711 module, a precision 24-bit analog-to-digital converter (ADC) specifically designed for weigh scale applications. The HX711 contains an internal programmable gain amplifier (PGA), which increases the sensitivity of the load cell signal, typically by a factor of 128 or 64, before digitizing it. This amplification is essential for achieving high-resolution measurements, as the unamplified signal from the Wheatstone bridge would otherwise fall below the noise floor of most general-purpose ADCs.

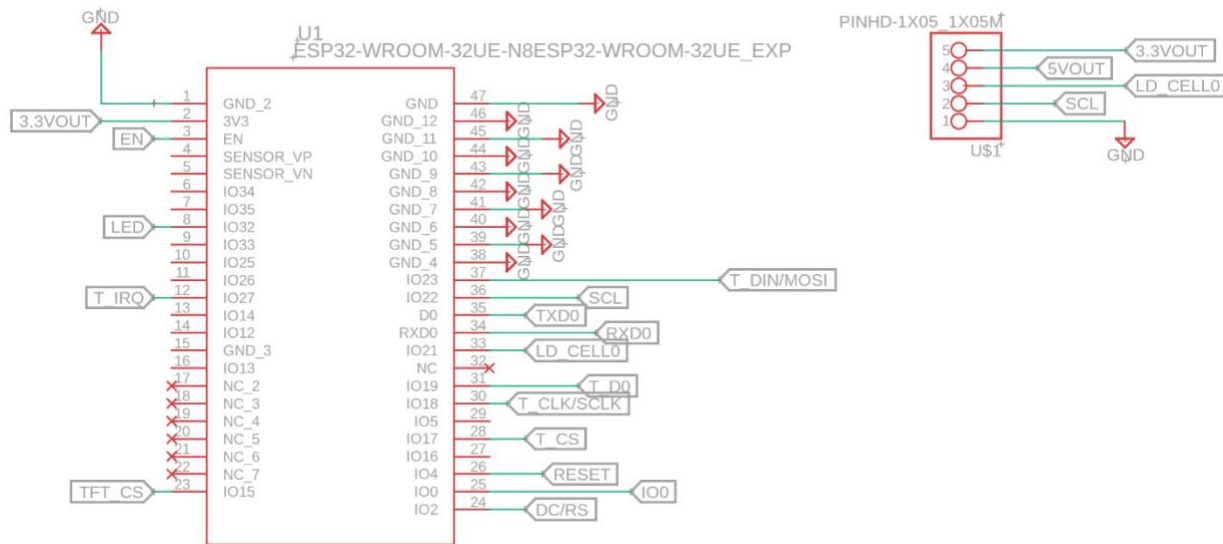
The ESP32-WROOM-32 microcontroller in our system is responsible for managing the data acquisition process. It generates the necessary clock pulses to initiate data transmission and

sequentially reads the output bits. After receiving the full 24-bit sample, the microcontroller can apply calibration coefficients, tare adjustments, or filtering algorithms as needed to process the raw data into a usable and accurate weight measurement. This processed data can then be displayed on a touchscreen, logged for future analysis, or used to trigger actions such as label printing.

In summary, this architecture, starting with the Wheatstone bridge-based load cell, followed by signal amplification and analog-to-digital conversion via the HX711, and ending with real-time data processing by the microcontroller, ensures high-precision, reliable, and repeatable weight measurements. This system is well-suited for our portable digital scale application, where accuracy, low power consumption, and compact design are essential.

In our final design, we revised the load cell connection by wiring it directly to the load cell amplifier board, rather than routing it through the main PCB. This change was made to simplify the signal path, reduce potential noise or signal degradation, and improve overall reliability. Since the load cell produces low-level analog signals, minimizing the length and complexity of the wiring before amplification is crucial for maintaining accuracy. By connecting the load cell directly to the amplifier, we ensured cleaner signal transmission and reduced the chance of interference. This also allowed for a more modular setup, making assembly and troubleshooting more straightforward during testing and integration.

Figure 13: Load Cell Schematic



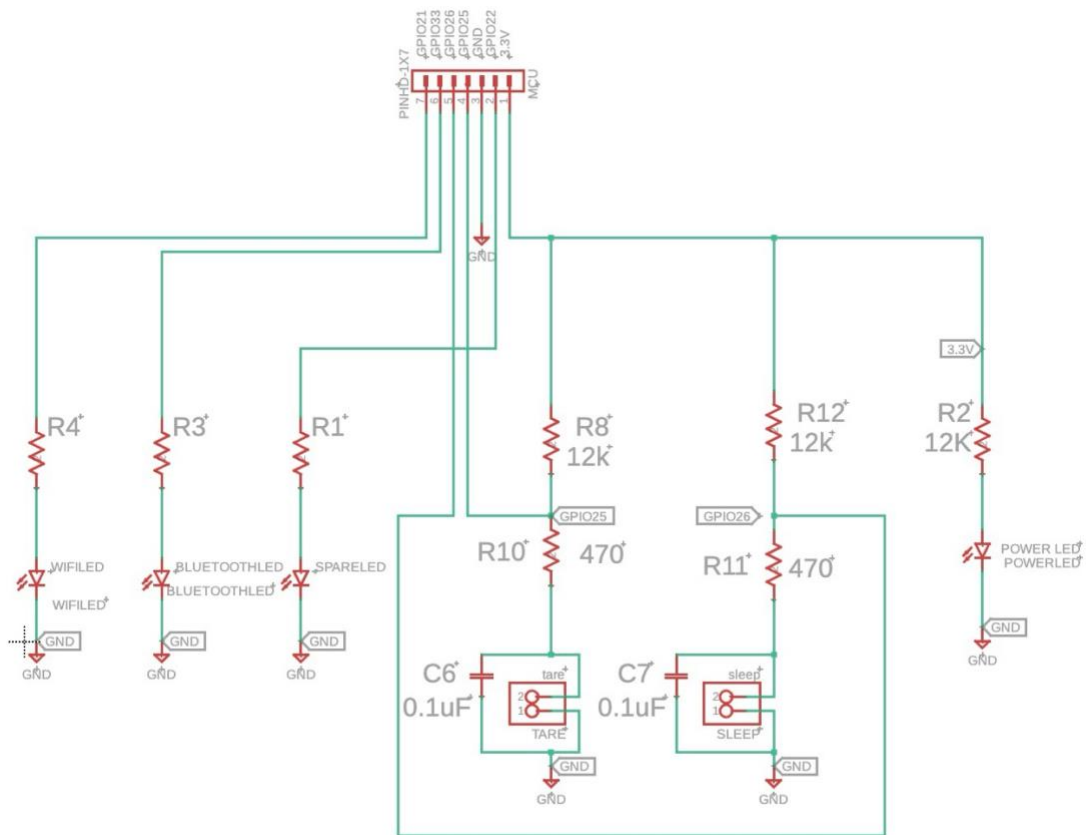
6.8 LED Schematic

We will be integrating LEDs into the design to provide real-time system status feedback. Two key indicators will be implemented: the Battery Low LED and the System Online LED. The Battery Low LED will light up when the system's battery voltage falls below a predefined threshold, alerting the user that the battery needs to be recharged or replaced. The System Online

LED will indicate that the system is powered on and functioning correctly, remaining illuminated when the system is active and performing as expected. These LEDs will be connected to the microcontroller and controlled based on specific system conditions, offering clear visual indicators of the system's status at any given time. The schematic provided illustrates how we plan to integrate the LEDs into our design once it is finalized.

In our final implementation, we updated the functionality of our LEDs and redesigned the daughterboard to include the required physical buttons. The LEDs now indicate Bluetooth, Wi-Fi, and system-on status. We also added dedicated tare and sleep buttons. These components were routed using wired connections that extend to the exterior of the enclosure for easy user access.

Figure 14: LED to MCU schematic



6.9 Printer

To establish reliable communication between the label printer and the microcontroller, we are incorporating a dedicated communication module that serves as an intermediary between the ESP32 and the printer. This module ensures signal compatibility and streamlines the data exchange process, especially given the differing electrical and protocol requirements that often exist between microcontrollers and peripheral devices like thermal printers.

For this implementation, we are utilizing the I²C (Inter-Integrated Circuit) interface to handle communication between the ESP32 and the communication module. I²C is a widely used, two-wire serial communication protocol that enables efficient, low-overhead data transfer between integrated circuits. It is particularly well-suited for applications where multiple devices need to communicate with a central controller, as it allows multiple slave devices to share the same bus with a unique address assigned to each.

The communication module interprets commands from the ESP32, formats them according to the printer's protocol requirements (which may involve UART, parallel, or proprietary formats depending on the printer model), and forwards them to the printer accordingly. It may also handle tasks such as buffering data, managing flow control, and relaying printer status information, such as error notifications or print completion signals, back to the microcontroller. This setup provides a robust, flexible, and scalable solution for integrating the printer into the system without overburdening the ESP32 with additional protocol handling or signal conditioning tasks.

6.8.1 Printer to MCU Communication Module

To establish reliable serial communication between our ESP32-WROOM-32 microcontroller and the thermal label printer, we employ the MAX3232 RS-232 transceiver. This interface component is essential due to the fundamental difference in logic level standards between the two devices: the ESP32 operates using 3.3V TTL (Transistor-Transistor Logic) UART signals, while the printer communicates via the RS-232 standard, which uses significantly higher and inverted voltage levels (typically $\pm 12\text{V}$ for logic high and low). Directly connecting these two systems without proper level shifting could result in data corruption or hardware damage. The MAX3232 acts as a bidirectional voltage level translator, ensuring safe and effective signal conversion between the ESP32 and the printer.

The MAX3232 is particularly well-suited to this application due to its compatibility with low-voltage digital systems. It operates over a wide supply voltage range of 3.0V to 5.5V, allowing it to interface seamlessly with the ESP32's 3.3V logic. The device includes two line drivers and two receivers, enabling full-duplex communication, meaning data can be sent and received simultaneously. This is critical for applications like label printing, where command-response communication is often required to control print operations and receive status feedback. A key feature of the MAX3232 is its internal charge pump, which generates the high-voltage RS-232 signals required for communication with the printer from a single low-voltage supply. This charge pump requires four external 0.1 μF capacitors that are connected to specific pins on the IC to regulate the voltage doubling and inverting processes. These capacitors are typically arranged between the C1+, C1-, C2+, C2-, V+, and V- pins as per the standard application schematic provided by Texas Instruments. The use of a charge pump eliminates the need for multiple power rails, simplifying the power supply design of the system.

In terms of hardware connections, the ESP32's TX (Transmit) pin is connected to the T1IN (Transmitter 1 Input) pin on the MAX3232. This signal is level-shifted and output through the T1OUT pin, which is connected to the RX (Receive) line of the printer. Conversely, the printer's TX line is connected to the R1IN (Receiver 1 Input) on the MAX3232, which converts the RS-

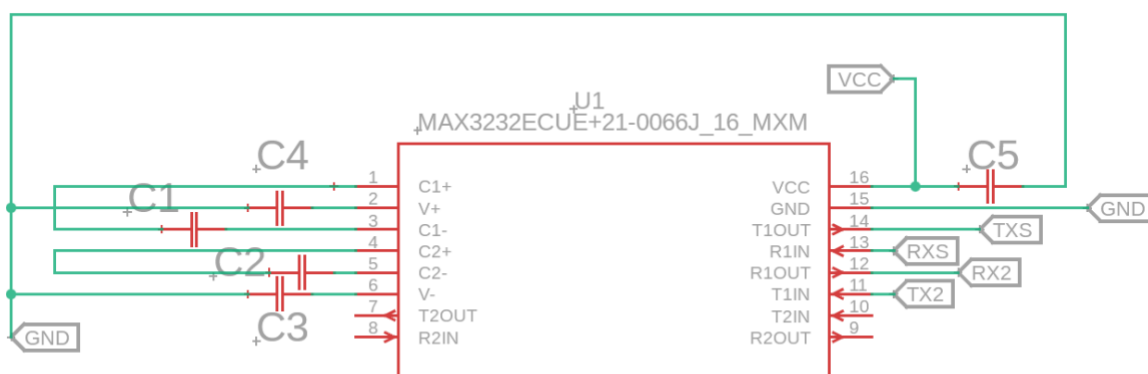
232 signal down to a TTL-level signal and outputs it on R1OUT, which is then connected to the ESP32's RX (Receive) pin. This arrangement ensures that both outbound and inbound serial data are properly translated, maintaining signal integrity across the voltage domain boundary. The VCC pin of the MAX3232 is powered by the ESP32's 3.3V rail, and the GND pin is connected to the common ground shared between the microcontroller, the transceiver, and the printer. Maintaining a shared ground is essential for proper operation and to prevent communication errors caused by floating voltage references.

Overall, the use of the MAX3232 in our system guarantees robust and compliant serial communication between the ESP32 and the RS-232-based printer. This integration supports reliable command transmission and data exchange, which are vital for real-time label printing operations. By leveraging a dedicated transceiver chip like the MAX3232, our design ensures electrical safety, voltage compatibility, and dependable performance in a compact and cost-effective solution.

In our final implementation, a level shifter was no longer necessary and therefore was omitted from the design. This decision was based on the fact that our selected thermal printer operates using TTL-level (Transistor-Transistor Logic) serial communication, which is directly compatible with the ESP32's 3.3V logic.

By eliminating the level shifter, we simplified the overall system design, reduced component count, and minimized potential sources of communication issues. This also helped conserve space on the PCB and improve power efficiency. The direct TTL compatibility ensured reliable data transmission between the microcontroller and printer without any signal conditioning.

Figure 15: MAX3232 to MCU schematic



Chapter 7 Software Design

From the software perspective, platform capability was a critical factor in ensuring accessibility. Our choice of programming language and development framework determined whether the mobile app would be Android-only, iOS-only, or cross-platform. We chose to develop using

Flutter, a cross-platform solution, which allowed us to maintain a single codebase for both platforms.

While native development—using Java or Kotlin for Android and Swift for iOS—would have provided optimal performance and full access to device-specific features, it would have also increased development time and required separate codebases. By opting for Flutter, we streamlined the development process and reduced ongoing maintenance efforts. However, this came with trade-offs, such as limited Bluetooth connectivity support and slightly reduced performance compared to native apps.

We implemented real-time data syncing alongside offline functionality, which added complexity, particularly when handling live weight data from the scale and ensuring dynamic inventory updates remained accurate. The chosen platform and framework directly influenced how the app was maintained, scaled, and updated throughout development.

As previously mentioned, we kept the application minimal by focusing on its core features. However, given the potential for future commercialization, we also designed with manufacturability, scalability, and long-term software maintainability in mind.

7.1 Mobile Application Development

The Pumped project aimed to create a portable scale and label maker for breastmilk with a mobile application designed to directly receive information from the device. When developing this application we needed to establish user requirements for the targeted market. The primary function of the planned application was to provide caregivers with a seamless way to measure, log, and label breastmilk storage.

In our original design, the device was intended to transmit breastmilk-related data to the mobile application via Bluetooth immediately after label printing. This would have enabled real-time weight management and automated label generation within the app. Planned features included editable milk logs, notifications for upcoming expiration dates, and predictive alerts for when additional milk might be needed. These features were envisioned to reduce manual input errors, improve labeling accuracy, and support better inventory tracking. By aligning these functions with user needs, we aimed to make the integration of the mobile application both intuitive and impactful.

While the original plan envisioned Bluetooth as the primary communication channel between the device and mobile application, the final implementation evolved for reliability and real-time data syncing. In practice, Bluetooth was used exclusively to transmit the user ID (UID) from the app to the device during the initial pairing process. All breastmilk measurement data, label information, and inventory logs were instead sent directly from the device to Firebase over WiFi. This approach allowed us to maintain accurate cloud-based tracking while minimizing the amount of Bluetooth communication needed.

Given that the target users were breastfeeding mothers, we prioritized ease of navigation and adaptive features to support a personalized experience. The user interface was planned with simplicity and accessibility in mind to ensure ease of use for all individuals. To accommodate

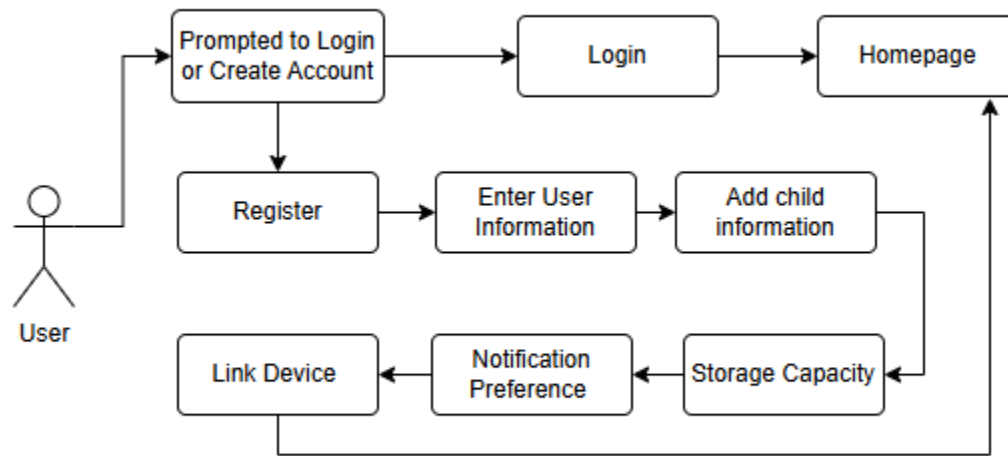
varying caregiver needs, we envisioned the application offering customization for labeling preferences, storage duration reminders, and feeding logs. Users would be able to create separate profiles for each infant, enabling more personalized tracking of milk intake and storage history.

The application included a history log that enables users to keep track of past measurements and storage. Utilizing this data, users received notifications about upcoming milk expirations, low inventory alerts when more milk may be needed, and easy access to storage history and expiration dates. These features were meant to promote adherence to best practices in milk preservation. Overall, by designing around user needs and expectations, we aimed to improve the efficiency and convenience of milk tracking and labeling through a thoughtful and well-integrated software experience.

7.1.1 Mobile Application

Our mobile application followed a structured workflow to ensure seamless logging, inventory management, and label printing. The process began when a user opened the application and encountered the login screen. The user entered their credentials and, if correct, was directed to the milk log homepage. If the credentials were incorrect, they had the option to reset their password. Upon selecting this option, the user was taken to a verification page where they must enter their email address. An email is then sent with a verification link that the user must click to confirm their identity. After verification, the user can return to the app, change their password, and will then be prompted to log in again.

Figure 16: First Time Access to App Flowchart



Once logged in, the main page displays the milk inventory log. Below the logs, users were presented with various options to add, edit and delete logs, print labels, and modify application settings. Each function was designed to help maintain accurate inventory and track milk usage. To use the edit, delete, or print functions, the user had to first select the log they wanted to act upon. After each process was completed, the user was returned to the homepage. If the user wanted to cancel an action, they could navigate away using the back option. A description of the main functions from the flowchart is provided below.

Figure 17: App Functionality Flowchart

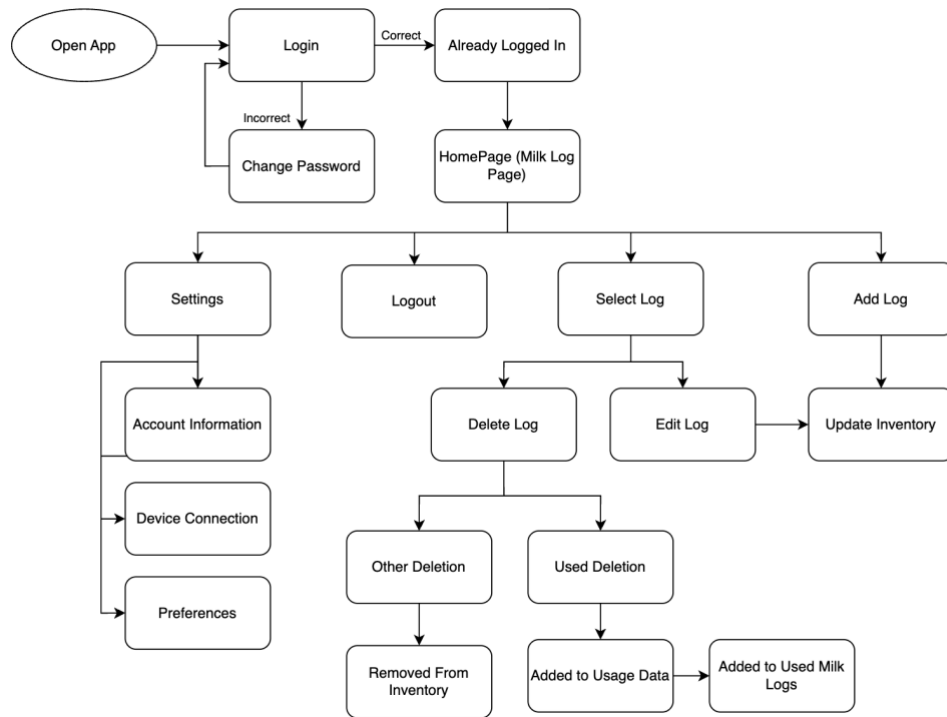


Figure 18: App Prototype

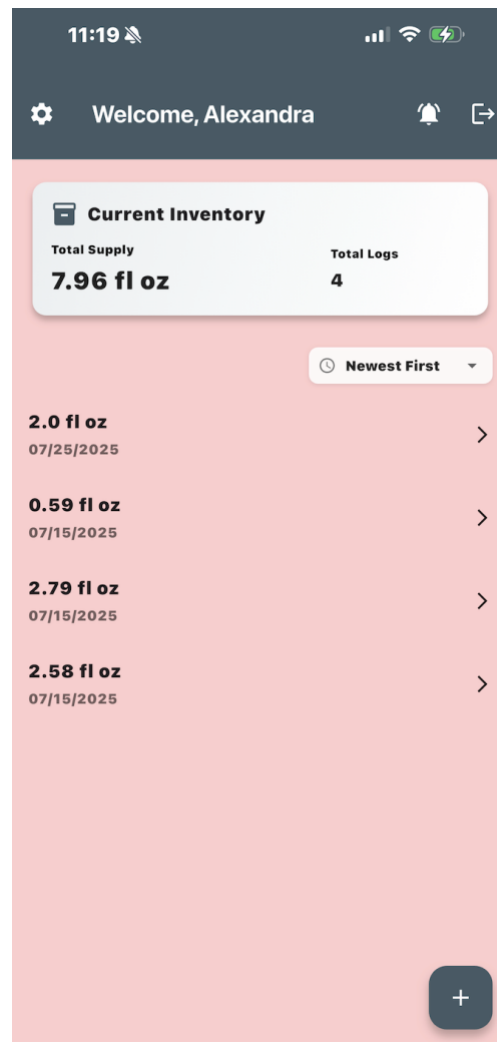
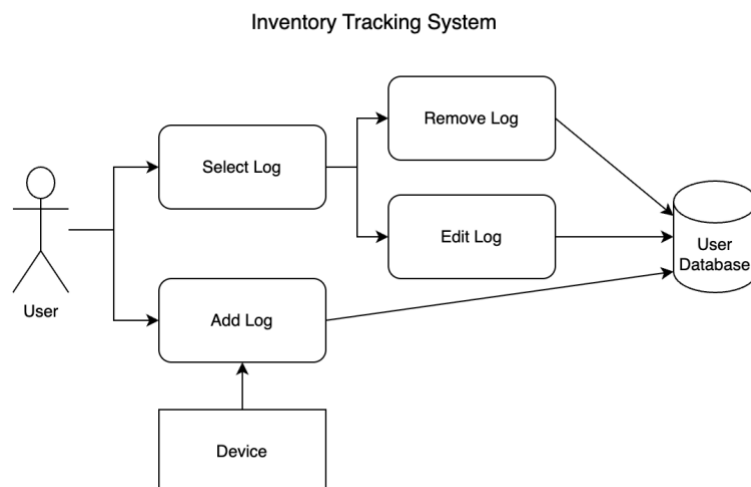


Figure 19: Inventory Tracking System Flowchart



Add_log Function:

After selecting the Add option, users were redirected to a new screen that prompting them for date, time, volume, content type, storage method, and allergens. Dietary details were selected via checkboxes, and optional notes could be added. The log was submitted to the Firebase Realtime Database under the authenticated user's UID and shown in the inventory list.

Edit_log Function:

This function was intended to work similarly to the Add option. After clicking Edit, users could update volume, date, time, storage type, allergens, and notes. Once saved, changes were written directly back to the user's log entry in Firebase. Navigation returned the user to the updated homepage with confirmation feedback.

Delete_log Function:

This function was planned to require two steps. After selecting a log and tapping Delete, users would see a confirmation page showing the log's details. A back button was available at every step. Users were then asked whether the deletion was "used" (to be tracked in the usage database) or "other" (mistaken entry or expired milk). Only "used" deletions would be included in future milk need calculations, hence the importance of the data being stored accurately. Deletions removed the log from the user's inventory list and triggered inventory recalculation if applicable.

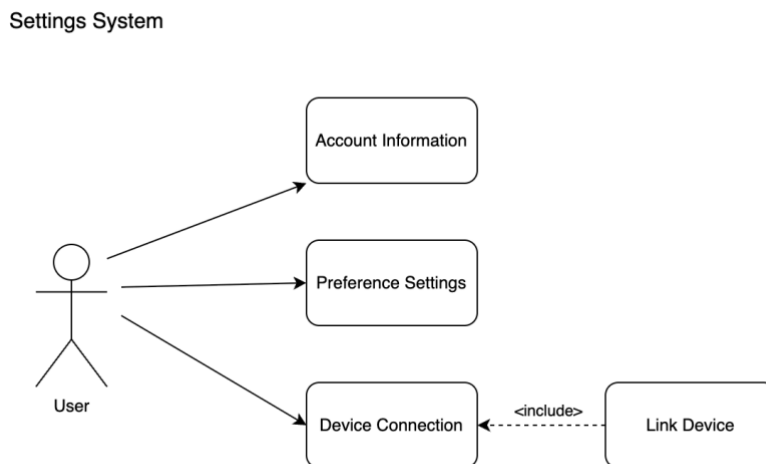
Print_log Function:

During development, we originally planned for the user to initiate label printing directly from the mobile application. However, in the final implementation, printing was handled entirely on the embedded device's user interface rather than through the app. The mobile application did not transmit log data back to the ESP32 via Bluetooth. Instead, all logs were stored in Firebase, and printing occurred locally on the device once the scale detected a stable weight. The app's role was limited to sending the user's UID to the device, allowing the ESP32 to correctly associate printed labels with the appropriate user, but the print command itself was not triggered from the app.

Settings Function:

The Settings function was the most feature-rich component of the application, containing multiple subsystems. Upon selecting the Settings option, the user was presented with a dedicated page displaying several sub-options: Account Information, Preferences, and Device Connection. Each of these sections allowed users to view or adjust aspects of their app experience, as described below.

Figure 20: Settings System Flowchart



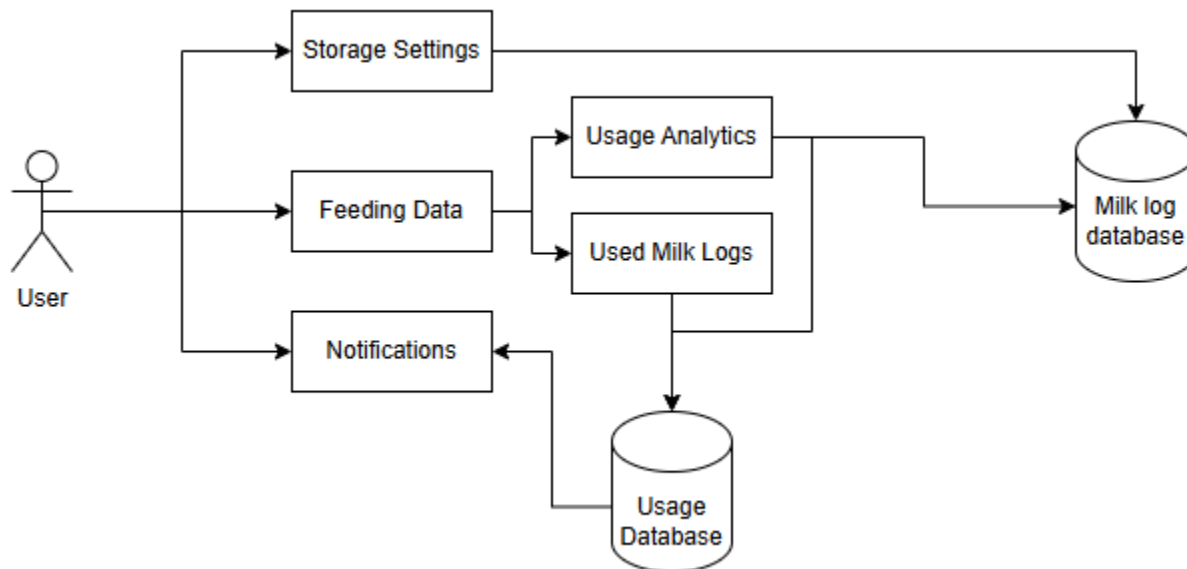
Account Information:

When users selected the Account Information option, they were taken to a page displaying personal details such as their username and the email associated with their account. This page did not allow users to directly edit their information. However, if the “Change Password” button was selected, a password reset email was sent via Firebase, using the same flow as the login screen’s reset option. While the original design included a section for entering child-related details such as age, gender, weight, and typical milk consumption, this feature was not implemented in the final version. Instead, basic child information was collected during the user registration process.

Preference Settings:

The Preference Settings section allowed users to personalize their app experience, primarily by customizing storage and notification preferences. While the initial design included options for dietary restrictions and feeding schedules, these were not implemented in the final version. Instead, users could adjust how long milk could be stored under different conditions and configure alert settings. In the notifications section, users selected the types of alerts they wanted to receive—such as expiration reminders and future milk need predictions—and could define when those notifications should occur. For example, a user could set an expiration alert to trigger one month after a milk log’s date to reflect their personal storage guidelines.

Figure 21: Preference Settings Flowchart



General Settings:

In the final implementation, general settings were streamlined and combined with the device connection functionality. A section labeled “Link Device” was included as a placeholder to represent device pairing. Actual device connection was managed through Bluetooth Low Energy (BLE), where the mobile application transmitted the user's UID to the embedded device. Features like language selection or detailed alert controls were part of the original plan but were not implemented.

Logout:

Selecting the Logout option on the main page securely signed the user out of their account and redirected them to the login screen. This cleared the active session and ensured authentication was required upon reentry to the app.

Table 19: Function and Variable Chart

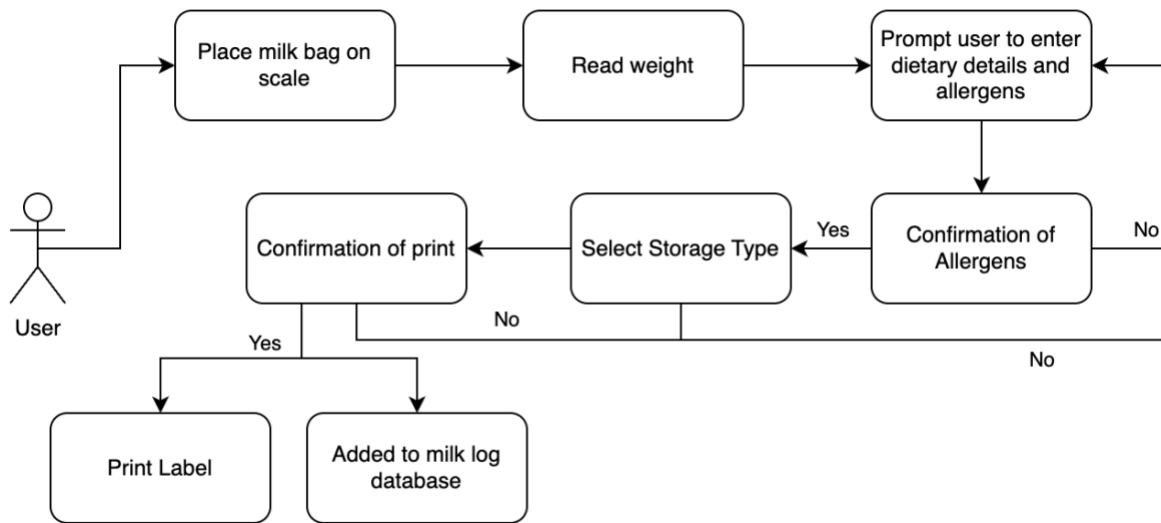
Label	Function or Variable	Description
Add_log	Function	Users entered essential details such as volume, date, time, storage type, content type and dietary information. Once data is confirmed the log is added to inventory and user database.
Edit_log	Function	Allowed users to modify existing milk log entries including volume, date, time, storage type, content type and dietary information. Updated logs are saved in the user database.
Delete_log	Function	Users review log details before selecting either Used Milk Log Deletion or Other deletion.

Settings	Function	Provides access to account information, preferences, Device Connection.
Log	Variable	Represents an individual milk entry containing all relevant data such as volume, date, time, storage type, content type, and dietary information.

7.2 Device Software

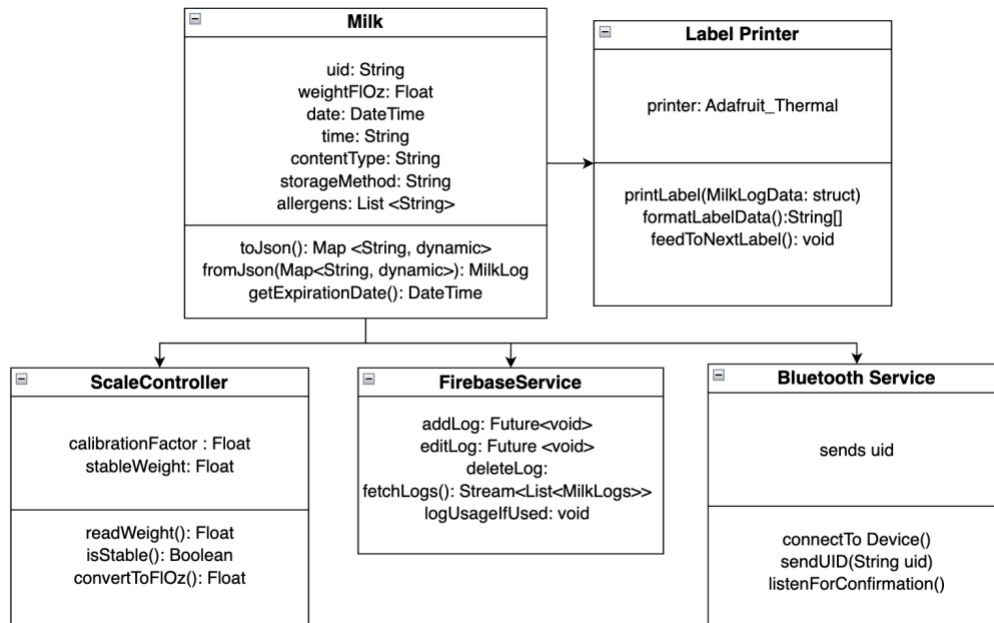
The software architecture of the device was built around an ESP32-WROOM-32 microcontroller, which managed input, output, and communication across all peripheral subsystems. The firmware operated using a loop-based execution model that relied on event flags and state monitoring to ensure responsive user interaction and reliable hardware control. As initially planned, the typical use case began with the user placing a breastmilk bag on the scale, where the device measured the weight using the load cell system and waited until a stable value was detected. The user was then prompted on the LCD touchscreen to enter dietary information, which would appear on the printed label. After entering the relevant details, the user had the opportunity to review and confirm the data. If any information was incorrect, they were redirected to confirm or revise the selected allergens. Once the data was verified, the user proceeded to a screen where they selected the intended storage method—such as fridge, freezer, or room temperature, which informed expiration tracking. Following this, the system displayed a final print confirmation screen. Although the original design intended for the label data to be transmitted to the mobile application via Bluetooth after printing, the final implementation differed. Printing was handled entirely on the embedded device, and Bluetooth was used only to transmit the user's UID during the initial pairing process. All milk log data, including volume, content type, allergens, and storage method was sent from the ESP32 to the Firebase Realtime Database via WiFi. This approach ensured that logs appeared in the mobile application without relying on Bluetooth data transfer. If the user chose not to print, the interface returned to the dietary input screen, allowing them to revise or restart the process as needed.

Figure 22: Use Case Diagram



Each subsystem was organized into modular software components to promote clarity, reusability, and future expandability. The diagram below illustrates the high-level class relationships within the system's architecture. At the center was the Milk class, which encapsulated key information including weight, expiration date, allergen presence, and storage type identifier. This class was associated with the Weight and Storage Condition classes, which stored detailed attributes such as volume, label size, storage environment (e.g., fridge or freezer), and the corresponding maximum storage duration. The Label Printer class interacted with the Milk class to generate and print formatted labels containing relevant metadata such as date, time, and weight. Together, these classes modeled the complete flow of a milk log from measurement to labeling and storage tracking.

Figure 23: Software Class Diagram



7.2.1 Load Cell Subsystem

For our project, we used an HX711 load cell amplifier to process weight measurements from the load cell. The HX711 amplified the small analog signal from the load cell and converted it into a digital signal, which was then read by the ESP32-WROOM-32. Although our initial plan specified I²C communication, the final implementation utilized the HX711's default two-wire interface (DT and SCK) to interface with the microcontroller.

To ensure the system operates safely, we included basic overload detection to prevent the system from exceeding the scale's weight capacity. The firmware continuously monitored readings from the HX711, and if a value above the safe threshold was detected, an error screen was displayed on the touchscreen to alert the user. While additional safety features such as audible alerts or system lockout were considered, they were not implemented in the final prototype.

This setup ensured stable and accurate weight readings while protecting the hardware from excessive load. Weight stabilization logic was also implemented in firmware to confirm that a reading had settled before allowing the user to proceed to the next steps.

7.2.2 LED Subsystem

In our final design, LEDs were used as system status indicators and were each connected to a dedicated GPIO pin on the ESP32. The LEDs operated in an active-high configuration, where setting the pin HIGH turned the LED on and setting it LOW turned it off. Each LED corresponded to a specific system status, including Bluetooth connectivity, WiFi status, and power status.

To manage the LEDs efficiently, the firmware included a dedicated function that updated LED states based on global system flags. These flags were updated as system conditions changed, and

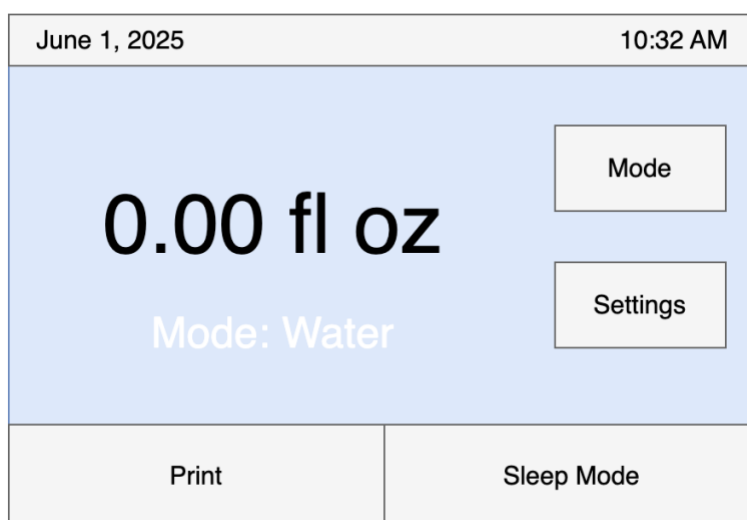
the LED update logic was only triggered when a change in state was detected. This approach minimized flickering and reduced unnecessary power consumption by avoiding continuous polling in the main loop.

7.2.3 Touchscreen Subsystem

The graphical user interface (GUI) served as the primary method of user interaction and was implemented on a 2.5-inch ILI9488 TFT touchscreen. It guided users step-by-step through the breastmilk labeling and logging process, from weight detection to allergen input and label printing.

The interface was designed with clarity and usability in mind, using large fonts, intuitive buttons, and sequential screens. It also visually displayed real-time system statuses such as current weight, Bluetooth connection, and battery level. During development, particular attention was paid to ensuring that the interface flowed smoothly and provided immediate feedback, including confirmation prompts and error messages when necessary.

Figure 24: Homepage Draft

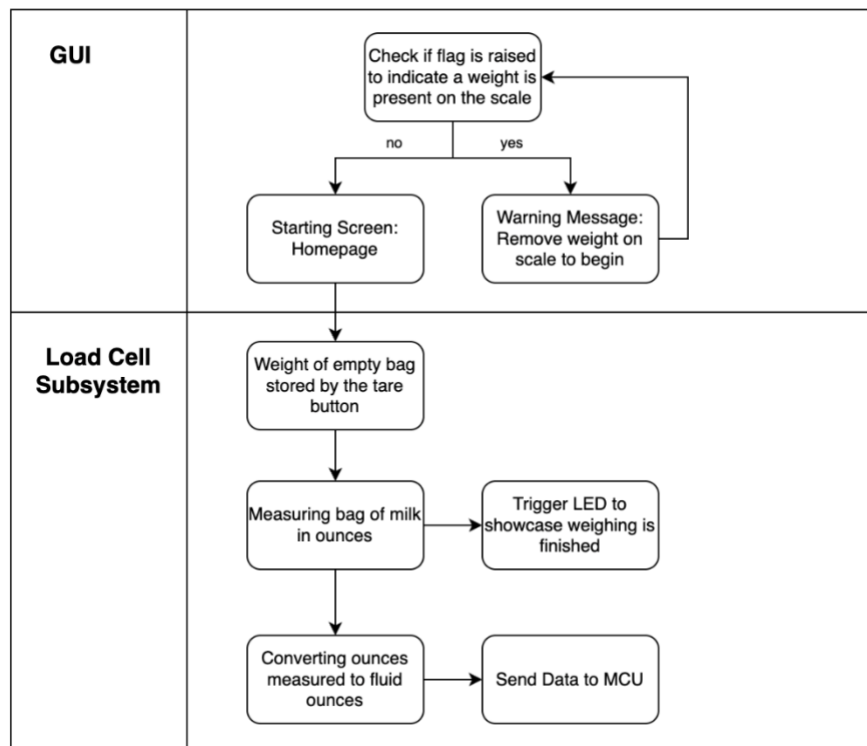


The primary function displayed on the device's home screen was the print feature, which allowed users to generate a label containing the date, time, milk volume (in fluid ounces), and any selected dietary or allergen information. In addition to printing, the home screen provided access to other essential features: Mode, Settings, and Sleep Mode. The Mode button enabled switching between different unit conversions for weight display. The Settings menu included options for Bluetooth pairing, date/time configuration, scale calibration, screen brightness adjustment, and viewing printer status. The Sleep Mode function placed the device in a low-power state to conserve energy.

While the initial design included a Reset button intended to clear the displayed weight, this feature was removed in the final implementation. During testing, it was discovered that the Reset function relied on behavior tied to USB communication with a laptop, and would not function reliably in standalone, battery-powered use. As a result, the system was redesigned to

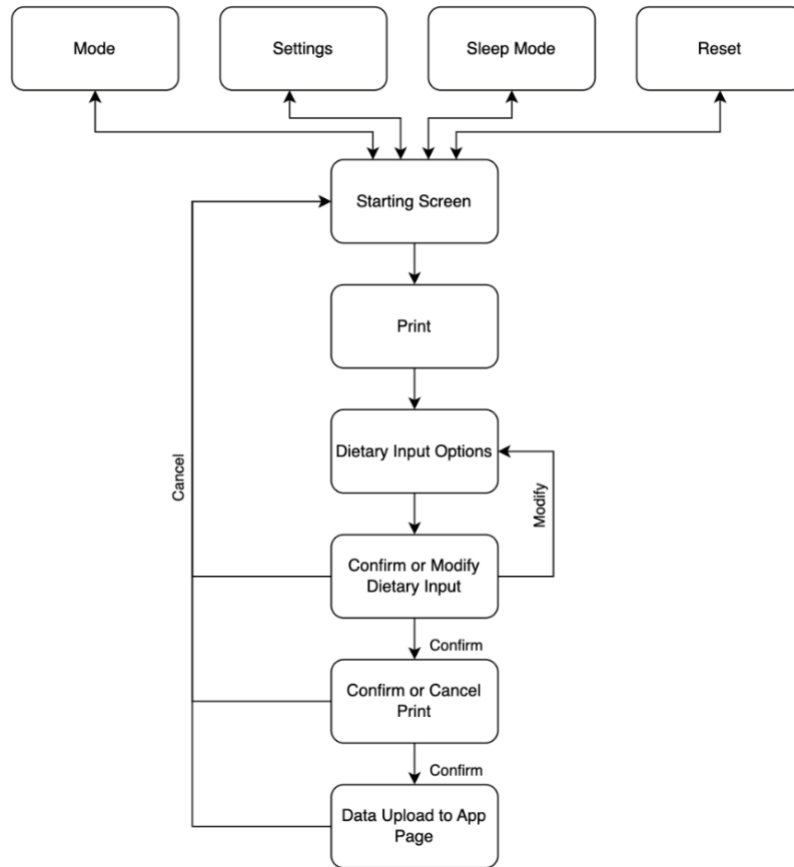
automatically clear and reset state variables after each completed print session, eliminating the need for manual resets by the user.

Figure 25: GUI and Load Cell Subsystem Integration



The weight reading, measured using the HX711 load cell amplifier, was prominently displayed on the home screen. Once a stable, non-zero weight was detected, a flag was set to indicate that an object was present. This flag was used to manage GUI behavior, such as enabling dietary input and progressing to print confirmation. If the flag was raised prematurely, such as when weight was detected before the tare process, an error screen instructed the user to remove the object and retry. Once the tare process was completed and a weight was added, the system allowed the workflow to continue. The GUI then guided the user through entering dietary and allergen notes, selecting the storage method, and confirming the print. After final confirmation, the label was printed and the log data, including volume, allergens, storage type, and timestamp was uploaded to Firebase via WiFi.

Figure 26: Touchscreen Flowchart



The touchscreen interface followed a multi-step confirmation process to reduce user error and guide the user through the labeling workflow. After the print function was selected from the home screen, users were taken through a series of screens to select and confirm dietary or allergen notes to be printed on the label. Once dietary input was confirmed or modified, the user was prompted to choose a storage method. They were then directed to a final screen where they could confirm or cancel the print. This structured approach ensured that all relevant information was accurately captured before printing, reinforcing a user-friendly and error-resistant experience.

To further reduce the likelihood of incorrect input, the touchscreen was programmed to recognize touch only within clearly defined regions associated with actionable buttons. Touches outside of these active zones were ignored, minimizing unintended selections. Upon final confirmation, the label was printed and the data recorded from the session was uploaded to Firebase, completing the workflow and synchronizing it with the mobile application.

7.2.4 Printer Subsystem

In the final implementation, the ESP32 managed communication with the thermal printer via a UART (serial) interface, not I²C as originally proposed. After the user completed the labeling

process through the touchscreen interface, the ESP32 assembled a formatted string containing the date, time, weight, allergen indicators, and storage type. This string was transmitted over UART to the printer, which generated a thermal label based on the received data.

All printing functionality was handled entirely by the embedded firmware. Although the original concept included Bluetooth-based transmission of print data from the app to the printer, this architecture was changed to keep printing fully local to the embedded system. Bluetooth was instead used solely for one-time pairing to transmit the user's UID to the ESP32. After each successful print, the ESP32 used WiFi to send the complete log entry to Firebase Realtime Database, allowing the mobile application to reflect the updated inventory in real time without needing to manage the print command itself.

This approach reduced system complexity, improved reliability, and ensured real-time database synchronization while preserving the portability and independence of the device. By leveraging UART for printer communication and WiFi for cloud integration, the device achieved efficient and accurate label printing in a compact embedded workflow.

Chapter 8 System Fabrication/Prototype Construction

In this section, we outline the process of constructing our prototype, which includes both the fabrication of the custom PCB and the development of a physical enclosure for the system.

The PCB was assembled based on our finalized schematic and layout, ensuring proper signal routing, power distribution, and component placement. We were mindful of soldering capabilities and used connectivity testing to ensure reliable performance during operation. We also verified compatibility between the microcontroller, load cell, amplifier module, and other peripheral components.

Simultaneously, we also designed and built a custom enclosure to house the electronics and mechanical components of the system. The enclosure was designed with usability, portability, and durability in mind, providing protection for the internal circuitry while allowing access to the touchscreen interface, label printer output, and necessary ports. Materials were selected based on weight, thermal characteristics, and structural integrity.

8.1 PCB Layout

Our PCB layout used a centralized PCB layout which brings together the electrical components of our system, including our DC/DC converters, MCU, pin headers for our load cells, pin header for the touchscreen, USB to UART bridge, and the printer communication module. We will also be including the battery charging module on the PCB, but we are considering making another unit for it due to the fact that it uses wall power. Separating the charging module into its own PCB would enhance safety, specifically in cases where power isolation is preferred.

Currently our PCB is an early prototype for the purpose of proving our technical understanding, though we intended on refining the design to be more compact and accurate to the needs of our product. The layout serves as a proof-of-concept to identify issues before the final fabrication.

We plan on using improving on our current board to achieve a more compact and efficient design that better aligns with the portability goal of our product.

On occasion, we use 6-millimeter traces for denser components with less space for wiring to avoid overlap. Overall, though, our PCB features 12-millimeter trace widths for better durability in carrying current without overheating, a concern for a project like ours with many different voltages required. Component placement was carefully considered to minimize cross-interference between nets, especially near analog sections such as the load cell amplifier.

In the future, we hope to implement better ground plane optimization, improved decoupling for noise-sensitive components, and increased separation of high-current and signal traces. Overall, this PCB forms the backbone of our embedded system and reflects the integration of both analog and digital design principles in a single, unified hardware platform.

Figure 27: Main PCB

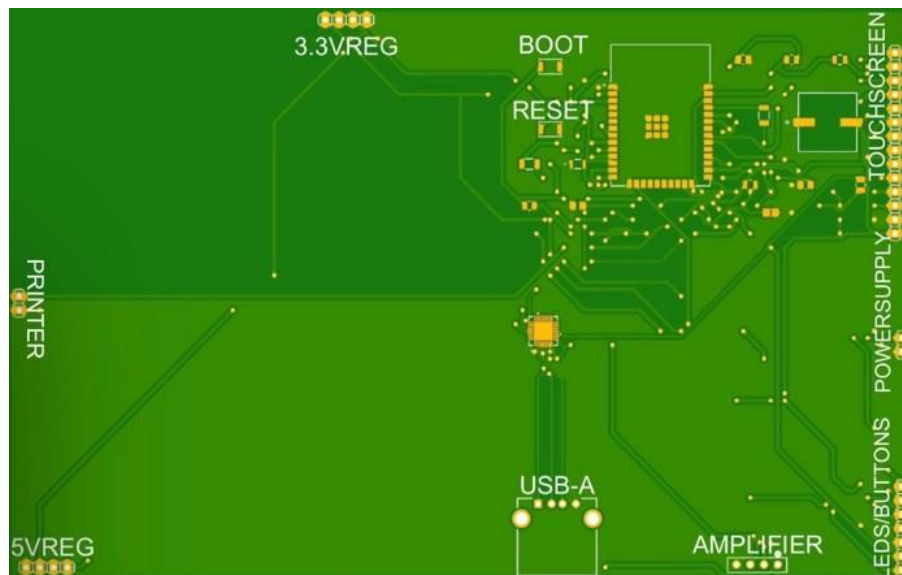


Figure 28: 3.3/ 5V Regulator PCB

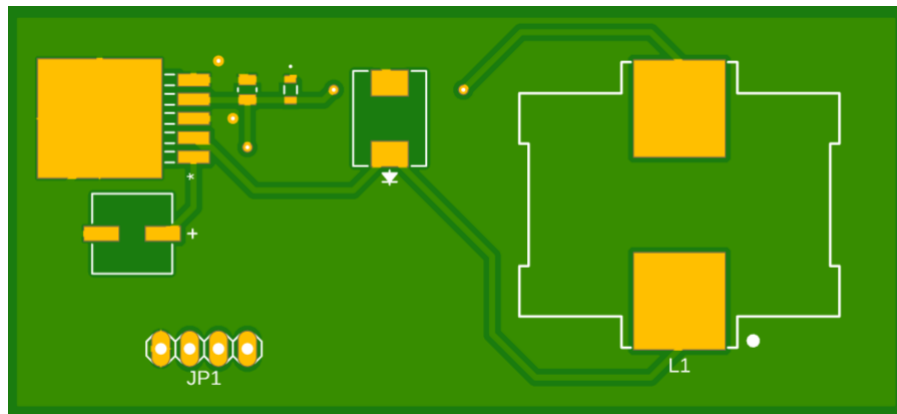


Figure 29: Load Cell Amplifier PCB

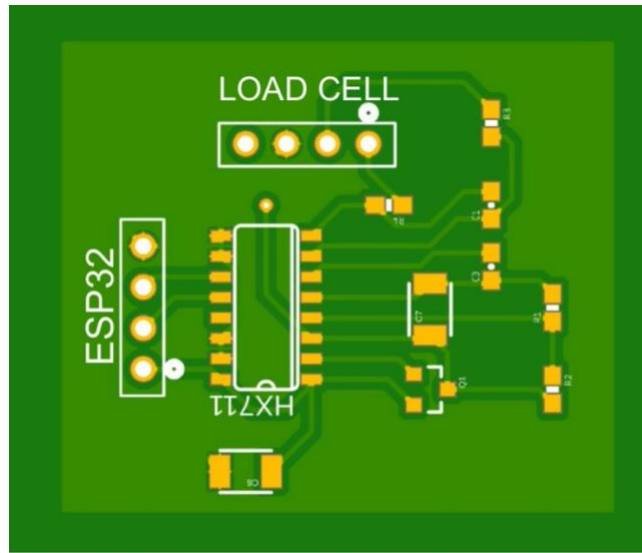
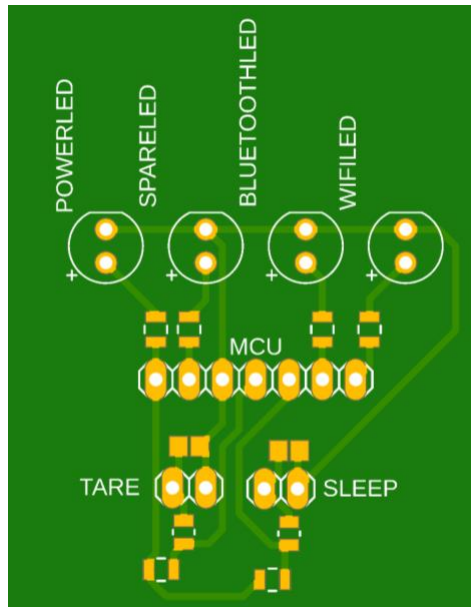


Figure 30: LEDs/ Buttons PCB



For our finalized design, we organized the PCB layout into one centralized main board and four separate daughterboards to simplify soldering and assembly. The main board housed the ESP32 microcontroller, USB-to-UART bridge, and pin header connections. The daughterboards were designated for the 3.3V voltage regulator, 5V voltage regulator, load cell amplifier, and the LED/button interface. Our LED / buttons daughter board used wires that allowed us to mount all our status lights and buttons externally.

8.2 Enclosure Prototype

The provided diagram is a hand-drawn mockup of the enclosure for the physical hardware. It is not drawn to proportion but provides a general layout for what was eventually built in Fusion. The enclosure mockup is roughly 5" (width) x 9.75" (length) x 2.5" (height) to accommodate the

dimensions of the hardware components that would've been embedded within it. Each dimension was selected to balance compactness with adequate internal space.

The scale, which is the widest component at 4", informed the total width of 5". The extra inch allows for clearance, mounting tolerance, and space for a raised lip to prevent the breastmilk bag from tipping and spilling.

The touchscreen display is the tallest component at 2.75 inches. To provide a better user experience, we've added approximately 1 inch of extra height for installation clearance, cable routing, and structural support, bringing the maximum display height to 3.75 inches. However, due to the display's 50° front-facing tilt, the vertical height of the overall enclosure will remain at 2.5 inches. This angular design helps reduce the vertical profile of the device while preserving ergonomic visibility and touch interaction.

The internal length requirement is based on the combined size of the scale and thermal printer, which span approximately 6.25 inches. We've extended this section to 7.25 inches to include sufficient space between components and enclosure walls. An additional 2.5 inches is added to accommodate the slanted front face of the display, leading to a total length of 9.75 inches.

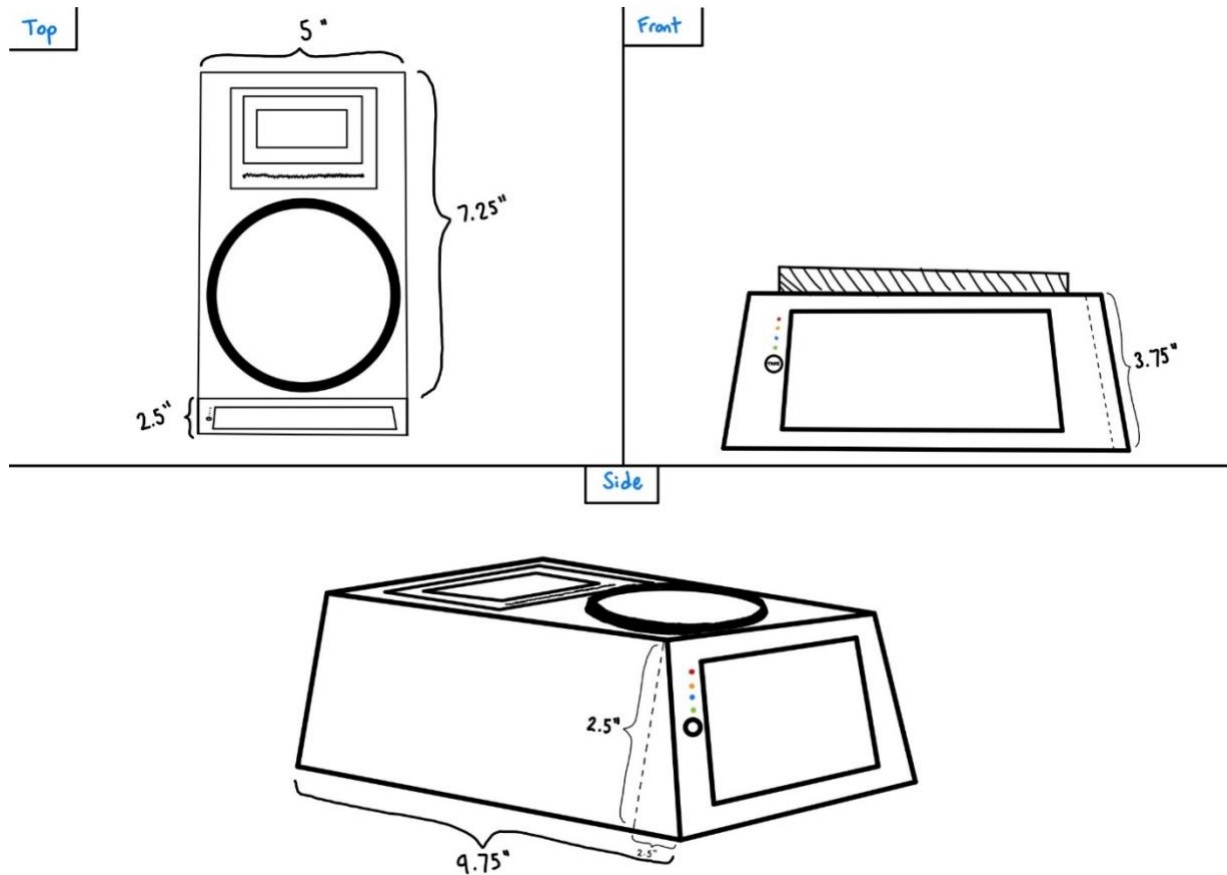
For the touchscreen display, we evaluated various orientations to determine the most intuitive and comfortable interface. A top-down, flat orientation proved restrictive and awkward to use, while a vertically upright configuration was less ergonomic for touch navigation. Our chosen solution is a 50° angled front face, providing an optimal middle ground that enhances visibility from both standing and sitting positions while maintaining ease of touch access.

A significant physical design feature is the raised circular guide surrounding the scale. This lip is intended to help stabilize the breastmilk storage bag during weighing, especially since the flexible and often unbalanced nature of the bag poses a tipping hazard. The guide will be modeled as part of the enclosure top plate and shaped to match the footprint of standard milk storage bags, ensuring both safety and repeatable positioning on the scale platform.

The final hardware consideration we implemented was the positioning of the thermal printer. For the easiest access to the paper tray for refills, we kept the printer face up at the top of the device. Since the height of the device is greater than that of the printer, we will implement a physical cutout of about one inch underneath the printer to support it and hold it upright. The direction of the printing slot is also positioned in a way that faces the user upon printing. This slot is positioned at a comfortable angle relative to the user to simplify label retrieval without interrupting the weighing or data logging process. The slot's alignment and surrounding lip helps prevent label jams or curling issues during printing.

This preliminary sketch serves as the foundation for our CAD model. In OnShape, this sketch was realized as a 3D model allowing us to simulate component fitting, test tolerance, and prepare for 3D printing. After being printed, the enclosure was tested again for real life fit with the hardware and was refined based on real-world feedback.

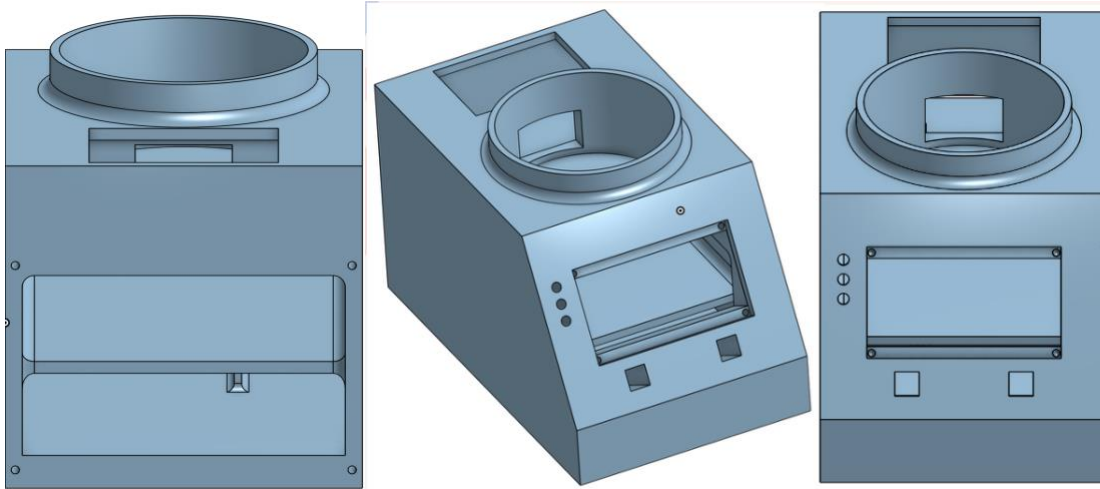
Figure 31: Enclosure Sketch



Throughout the course of SD2, we revised the design for our enclosure multiple times, adjusting for size & space logistics that were previously unaccounted for. Figure 29 shows the final CAD of our enclosure, and Figure 30 shows the final physically printed design fully integrated with the electrical components.

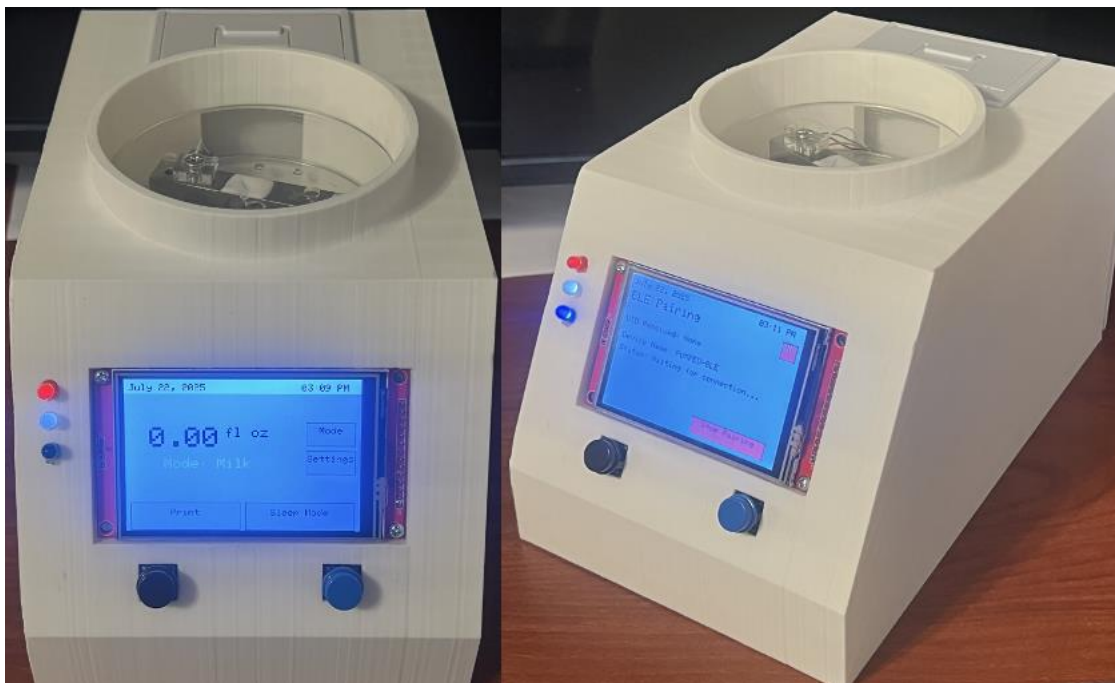
In our final CAD, there is an additional 90 degree bezel before the slant of the touchscreen, allowing for a sharper angle with the same overall length. Visible near the scale section of the design is a rectangular internal cutout that allows the wiring of the peripherals to be situated throughout the body in order to communicate with the MCU. The Tare (& Sleep) Buttons were moved to be below the screen, with the LED's remaining along the left side. In the back, the internals rest on a dual layer chamber, where the bottom layer holds the 12V DC Battery Pack and DC wires, while the top layer holds the PCB's. The bottom layer features a small column nearer to the righthand side which acts as a stabilizer to hold the battery in place while the device shifts around. This cutout for the internals is covered by a backplate that is held in place with 4 screw holes.

Figure 32: Final CAD



The final enclosure was printed using white PLA filament in the TI lab. The overall dimensions feature a 9.5" length, 5.75" width, and 5.4" height. The body was printed around 3 times in order to readjust the dimensions to fit the portability aspect as well as adding tolerances for the peripherals to fit into their designated slots. The buttons are glued in to their places, while the LED's were inserted with a rubber stopper in order to keep them from moving. The touchscreen is held in by 2 screws, and the printer/scale are resting unmounted in the enclosure.

Figure 33: Final Prototype



Chapter 9 System Testing and Evaluation

In this section, we will document all testing procedures related to both our software and hardware components. This includes breadboarding prototypes, verifying electrical connections, and performing initial functionality checks of individual modules such as the load cell, HX711 amplifier, microcontroller, touchscreen display, and label printer. On the software side, we will track our troubleshooting process, interface development, sensor calibration, and data handling routines.

The primary objective of this testing phase is to ensure that all components function as expected and that our overall system design is both feasible and reliable. By thoroughly testing each element early on, we aim to identify and resolve any hardware or software issues before full integration. This iterative testing approach will help us refine our design choices, improve system performance, and reduce the risk of complications during the final assembly and deployment stages.

Additionally, we will document any compatibility issues, unexpected behavior, or design revisions prompted by testing results. This record will serve as a critical reference point throughout development and will support future debugging, optimization, and version updates.

9.1 Hardware Testing

This section focuses specifically on the testing and validation of individual hardware components used in our project. Each component has undergone standalone testing to confirm its functionality, compatibility, and suitability for integration into the full system. This includes verifying electrical characteristics such as voltage levels, current consumption, and signal integrity, as well as checking for any physical or mechanical issues like loose connections or unstable mounting.

Testing has also involved evaluating how each component performed under expected operating conditions, including environmental factors such as temperature and load variations. By isolating components during this phase, we can detect early failures, identify out-of-spec behavior, and validate that each part meets its datasheet specifications before integration.

In addition, we have documented setup procedures, pin configurations, and any required adjustments to streamline future assembly or troubleshooting. This targeted hardware testing process is essential to building a stable and reliable final system.

9.1.1 Load Cell Hardware Testing

To ensure our system measures weight accurately and reliably, extensive testing was conducted. The components used to do so were the SparkFun 13329 load cell, SparkFun SEN-13879 HX711 load cell amplifier, and ESP32-WROOM-32. These components form the foundation of our weight measurement subsystem and ensuring their proper operation is critical to the success of the device.

Initial tests revealed that the load cell was either reading a constant or returning zero, which showcased it was not properly capturing different weight values. Our first assumption was a faulty connection between the load cell and the amplifier. Since we used a breadboard and the load cell amplifier did not have pins to connect it, we were using male to female jumper cables. Due to this we thought that loose jumper wires or sporadic contact might be causing the issue. We decided to solder jumpers to the load cell amplifier to allow for easier connectivity to the breadboard. After running multiple tests, we came to the conclusion that it did not resolve the issue either.

We then investigated whether the problem could be related to the connection between the amplifier and the ESP32. We started by checking our connections for the data (DAT) and serial clock (SCK) lines. After doing so we realized that they were connected to the incorrect pins for this communication and reconnected the amplifier to the correct GPIO defined in our code. Even after this connection we came to the realization that it still did not resolve the issue, so we started to do some more research into it. After doing so we found a forum where a lot of people were saying the load cell does not respond well to long wires which we had, but this also did not resolve the issue.

At this point, we turned our attention to the components themselves to verify whether they were functional. Using a multimeter, we measured the output voltages and resistances across the amplifier. The VCC had an output of 3.3 V while DAT and SCK had little to no voltage coursing through around 100 mV. The input and output resistance were supposed to be around 1000 ohms, which they were. The amplifier seemed to be functioning correctly from a power perspective, but we still couldn't get the expected output from the load cell. To further troubleshoot, we decided to compare our existing amplifier to an alternate one with similar functionality. We purchased a supplemental HX711 amplifier from Amazon since it would come quicker than ordering a new one from Digikey but still had the same functionality. To our surprise using the load cell we already had and our supplemental HX711 produced the results we expected and accurately detected changes in weight.

This confirmed that the issue was with our original amplifier. We suspect that during the soldering process, the board may have been overheated, or a solder bridge may have occurred, damaging its internal circuitry. It is also possible that repeated handling and wiring attempts compromised the amplifier's sensitivity. Moving forward, we are going to order more of the SparkFun HX711 module to test them and see if we have the same issue. Additionally, we intend to use header pins to make sure connections are easier to see and more secure. This experience reinforced the importance of testing each component in isolation, documenting behaviors, and validating with alternative hardware when possible. It also emphasizes how easy it is to pin problems to the wrong component and the need for methodical hardware troubleshooting.

During Senior Design II, we encountered issues while testing our amplifier, which we later traced back to a mismatch between our schematic and the actual development board. The problem was resolved by adding a ground connection directly to the schematic after the PCB had been fabricated.

9.1.2 Touchscreen Hardware Testing

Our next task after confirming our load cell and amplifier worked was to test the touchscreen. Due to delays caused by the theft of our original LCD screen, we used this period to test the microcontroller's communication and code functionality using an I2C IIC 1602 LCD Display Module. This allowed us to verify our ESP32's output capabilities while awaiting the arrival of the touchscreen we plan on using for our final design. Using the IIC 1603 LCD Display module's datasheet, we simplified the wiring process and ensured the proper pin connections. As an initial test, we uploaded a basic Arduino sketch that printed "ESP32 + LCD work Blinking LED on," which confirmed that the screen was correctly initialized and receiving serial data. We also connected an LED to indicate when the ESP32 was actively running the code, providing both visual and display confirmation of successful communication.

Once we received our ILI9488 LCD TFT touchscreen display, we were able to use the experience from the previous LCD test to begin hardware validation. We began by uploading a similar sketch to display a basic "Hello World" message, confirming that the screen was properly wired and powered through the ESP32, based on our reference schematics developed in Fusion360.

After establishing that the display was functional, we turned our attention to verifying the touchscreen functionality. A second Arduino sketch was uploaded to detect and display the X and Y coordinates of touch input in real time. This test confirmed the correct initialization of the touchscreen driver, as well as the ESP32's ability to register and interpret touches across different parts of the screen. The touchscreen consistently responded to user input, validating that both the display and touch layers were working as planned.

Following these tests, we integrated the wiring for the load cell and HX711 amplifier with the same ESP32 that was controlling the touchscreen. The goal was to verify that the microcontroller could simultaneously support both display and sensor operations. We modified our code to display real-time weight measurements from the load cell directly on the touchscreen, rather than displayed via the serial monitor. This integration required careful GPIO pin assignment and continuous compatibility checks between libraries to ensure stable performance. With this phase of testing complete, we confirmed that the ILI9488 touchscreen was not only functional in isolation but also fully compatible with other critical subsystems in our design.

9.1.3 Printer Hardware Testing

To begin testing our label printer, we first connected it to a regulated power supply to verify that it powered on correctly. Initially, we applied a voltage of 9V at 3A. Once powered properly, we confirmed that the device was functional and ready for further testing.

Next, we proceeded to evaluate the printer's output capabilities. While attempting to connect the printer to Windows-based computers, we encountered compatibility issues, specifically that the printer was not immediately recognized or was difficult to configure. However, we successfully established a connection using macOS systems, which allowed us to continue testing. Once connected, we were able to print test receipts and custom documents without issue, verifying the printer's core functionality.

Additionally, we confirmed that the printer is compatible with label paper, successfully printing on various adhesive-backed media formats. This confirmed that the printer is suitable for our application, which requires clear and consistent printing of weight data on labels. These tests provided a solid foundation for integrating the printer into the final system alongside the microcontroller and touchscreen interface.

In our final design, we selected a printer that communicates with the ESP32 using TTL-level signals, ensuring compatibility without the need for a level shifter. The printer operates at 12V, and we successfully verified its functionality by testing the RX and GND connections on a breadboard setup.

9.1.4 Power Supply Hardware Testing

To ensure that our power supply could reliably support each component in our system, we conducted a series of tests to verify voltage requirements and electrical connections. First, we reviewed the recommended voltage specifications for all key components, including the touchscreen, load cell amplifier (HX711), and label printer, to confirm their input requirements. We then used a multimeter to measure the output of our power supply and ensure it delivered consistent and accurate voltage levels that matched these requirements.

Special attention was given to the integrity of physical connections. We thoroughly inspected the power supply wiring and connectors to ensure there were no signs of damage, loose contacts, or potential short circuits that could affect performance or safety.

For the label printer, we provided a dedicated 9V power supply and verified the voltage output using precise measurements. During testing, we confirmed that the printer powered on and functioned as expected. Similarly, we verified that both the touchscreen and the HX711 load cell amplifier received appropriate voltages from their respective output pins, ensuring stable operation. These checks were essential in validating that the power system would support our components reliably throughout the project.

In the final integration of our project, we utilized a dedicated 12V, 3A power supply to reliably support components with higher voltage requirements. Additionally, we incorporated a regulated 5V supply to power lower-voltage peripherals and ensure stable operation across the system.

9.2 Software Testing

We began our testing with unit testing, isolating individual functions such as weight calculations, tare logic, label formatting, and data transmission to Firebase. These unit tests allowed us to verify that each software block handled normal and edge-case inputs appropriately. Following this, we conducted integration testing to evaluate how software modules communicated with each other and with physical subsystems like the load cell (HX711), touchscreen display, and thermal printer. For example, we validated that the ESP32 could detect stable weight values, allow user input through the touchscreen, and accurately transmit session data to the Firebase Realtime Database.

In Senior Design II, we placed significant focus on system-level testing under real-world usage. This included full end-to-end trials where a user would weigh breastmilk, enter allergen data, select a storage method, confirm print, and receive a printed label. At the same time, the log was uploaded to the app database and became visible on the mobile app. We tested for scenarios like premature weight detection (e.g., object placed before taring), user canceling mid-flow, accidental touchscreen input, power cycling during data processing, and disconnection during Bluetooth UID transfer. We implemented flag-based logic to catch these cases and ensure appropriate user feedback and recovery behavior.

To support this testing, we used serial debugging, on-screen LED indicators, and console logging to confirm system states. All test cases, behaviors, and resolutions were documented to guide future troubleshooting and potential product improvements.

9.2.1 Load Cell Software Testing

A key part of our software development was focused on the load cell subsystem, which was responsible for accurately capturing breastmilk volume through weight measurements. We used an HX711 amplifier to interface the load cell with the ESP32-WROOM-32.

We began with a basic code implementation to confirm that the HX711 amplifier was correctly wired to the ESP32-WROOM-32 microcontroller. In the initial version, we used the `scale.read()` function to return raw, uncalibrated data from the load cell. This test was used to verify there was proper communication between the load cell, amplifier, and microcontroller. We used GPIO pins 27 and 26 on the ESP32 for the data and clock lines respectively. The serial monitor output displayed consistent readings when weight was applied. Once that was established, we were able to move on to calibrating the scale.

Once basic communication was confirmed, we moved to calibration. We implemented the `scale.set_scale()` function, inputting a calibration factor that we determined using an iPhone 16 as a reference object. This involved averaging multiple weight readings and adjusting the factor until the output matched the expected weight in grams. We then converted these readings into ounces, and then the readings were updated in real-time and formatted to two decimal places on the LCD screen. As we continued to develop our software the scale was able to measure in different units based off user selection as opposed to manual code modification.

In our final implementation, we added a tactile tare button that users could press to reset the scale before weighing, which improved usability and accuracy. The scale was configured to detect when a non-zero weight was placed after taring and raise a flag for the GUI to prompt further user interaction.

During Senior Design II, we also implemented stability detection logic that ensured the system would not move forward with printing until the weight had stabilized within a tolerance range. This was crucial to prevent printing incorrect or mid-shift weight values.

These test routines were critical not only for verifying that the load cell hardware was functioning correctly but also for ensuring that the weight data formed a reliable foundation for downstream processes, such as label printing and log submission to Firebase.

9.2.2 Touchscreen Software Testing

The development and testing of the touchscreen interface played a central role in establishing a functional user experience for our device. Using an ILI9488 3.5" TFT LCD display, we began by testing its capacitive touch capabilities and interfacing it with the ESP32 microcontroller. One of our early software goals was to verify successful screen initialization, accurate detection of touch input, and real-time visual output on the display.

We began with a basic validation test using the `TFT_eSPI` and `XPT2046_Touchscreen` libraries. Our initial sketch displayed a simple message ("ESP32 + ILI9488 TFT") and printed X and Y touch coordinates to the serial monitor in real time. This confirmed correct wiring, SPI communication, and IRQ handling. Touches were consistently detected, and the screen reliably updated, verifying the functional integration of the display and touchscreen drivers with the ESP32.

Once we confirmed the touchscreen was operational, we focused on developing a more advanced graphical interface for the display. This included designing a homepage layout that featured a date and time header, a prominent section for displaying measured weight, and a series of button elements labeled "Mode," "Settings," and "Print." We expanded the software to continuously retrieve real-time weight data from the HX711 load cell amplifier and update the displayed value accordingly. This was achieved by integrating the load cell reading loop into the screen refresh logic, using `tft.fillRect()` to clear previous values and `tft.printf()` to print updated measurements in ounces with two decimal precision.

Additionally, we tested touch interaction by defining screen regions for future button-based functionality. While these buttons were not yet linked to any underlying logic, they provided the groundwork for upcoming interactive features such as screen navigation and command execution. This phase of touchscreen software testing validated the ESP32's ability to process sensor data and render real-time graphical output concurrently within a single program loop. It also demonstrated the touchscreen's reliability in both displaying the homepage interface and accurately registering user input.

The touchscreen software was tested extensively during labeling workflows. We confirmed that the display could guide the user through each step—from viewing live scale data, entering dietary and allergen information, selecting storage type, and confirming print. This multi-step confirmation structure significantly reduced the chance of user error and helped ensure accurate record-keeping.

Overall, the touchscreen testing validated that the ESP32 could handle concurrent touch input processing, sensor data acquisition, and real-time GUI updates. The interface proved to be stable, responsive, and capable of supporting the full user interaction flow required by our device.

9.2.3 Printer Software Testing

Printer software development focused on establishing stable serial communication between the ESP32 and a TTL-compatible thermal label printer to generate custom labels. Early in development, we encountered a compatibility issue with our initial printer, which used RS232 signaling incompatible with the ESP32's 3.3V TTL-level UART. To resolve this, we transitioned

to a different thermal printer that supported direct TTL serial communication, allowing us to proceed with firmware integration. Using the Adafruit_Thermal library and HardwareSerial on UART2, we developed a test script to initialize the printer and output formatted text. This script printed a sample label containing the date, time, and a mock milk weight, confirming that UART communication worked reliably and the printer could accept formatted strings.

Once the basic functionality was confirmed, we expanded the code to dynamically generate print content using real data collected from the load cell and touchscreen interface. During the final implementation, the touchscreen collected input from the user, such as dietary and allergen details, and storage type, then packaged this information along with the timestamp and weight measurement into a structured label format. After the user confirmed the data, the ESP32 sent the final formatted print string over UART to the printer for immediate label generation.

We verified the success of each print operation through physical observation and serial debugging. The final printed labels included the date, time, volume (in ounces), and user-entered notes. The printer consistently executed commands with minimal delay and no data corruption, demonstrating the reliability of our UART configuration.

In our final implementation, label printing was handled entirely on the embedded device, independent of the mobile application. This decision improved reliability and ensured functionality even without an active Bluetooth or WiFi connection at the time of printing. Printer testing was a key step in confirming that our system could deliver tangible, on-demand output to users. The ability to generate accurate, readable, and personalized labels was one of the core features of our project, and our tests ensured that it operated reliably in all tested scenarios.

9.2.4 Bluetooth Software Testing

Bluetooth communication was initially designed as a key feature for enabling connectivity between the embedded device and the mobile application. On the embedded side, our testing focused on establishing reliable Bluetooth Low Energy (BLE) pairing and correctly receiving the user's unique identifier (UID) from the mobile app to tag scale sessions with the appropriate account. We used the ESP32's built-in BLE stack to handle advertising, connections, and data exchange.

Importantly, our original design anticipated a more interactive BLE workflow—such as real-time log transfers or remote print commands from the mobile app—we ultimately decided to simplify the role of Bluetooth. In the final implementation, BLE was used only during the initial pairing process to transmit the UID. Afterward, all communication between the ESP32 and the Firebase Realtime Database was handled over WiFi.

This minimal but effective BLE integration allowed us to maintain secure, user-specific data association while keeping the embedded system logic streamlined and robust. The BLE software testing validated that our embedded device could pair reliably with a phone, receive user-specific information, and retain it throughout the logging process, ensuring accurate backend integration.

9.3 Plan for Senior Design II

To ensure that our project remains on schedule and achieves the desired functionality, we have developed a comprehensive and structured plan for our workload in Senior Design II. This phase marks a critical transition from conceptual design and simulation to the actual physical construction and integration of our final product. Our primary focus will be to assemble all hardware components, finalize the custom enclosure, and thoroughly test the system, ensuring that everything functions as expected.

A key objective for this phase will be the fabrication and testing of our custom PCB layout. This involves not only ensuring that the physical PCB is correctly manufactured but also rigorously verifying its performance. We will pay particular attention to key factors such as signal routing, power distribution, and the overall compatibility with all connected components. Verifying that the board performs as intended will be essential to the reliability and stability of the final system. Any issues discovered during this stage, whether related to trace routing, component placement, or power issues, will be promptly addressed to avoid delays later in the project.

In parallel with the hardware development, we will continue software development, allowing us to integrate and test features incrementally. By working on hardware and software concurrently, we can identify and resolve potential issues as they arise, rather than waiting until the hardware is fully assembled before testing the software. This parallel approach also allows us to fine-tune the user interface and ensure smooth communication between the various hardware components, ensuring that the system functions cohesively. Our approach will also include debugging and optimizing the software for the finalized hardware setup, paying special attention to aspects like real-time performance, sensor calibration, and communication protocols.

Furthermore, the enclosure design will be finalized to ensure that the components are securely housed and adequately protected from environmental factors. The enclosure will also be designed to facilitate ease of access for testing and troubleshooting, ensuring that the system is both functional and user-friendly.

Throughout this phase, we will also plan for thorough testing at multiple stages. This will include functional tests to ensure that the system meets all performance criteria, as well as stress and reliability tests to verify the system's durability over time. Additionally, we will perform iterative evaluations to detect and correct any problems early, preventing issues from compounding and ensuring a more efficient path toward finalizing the project.

By working on hardware assembly and software development in parallel, we can ensure that each part of the system is optimized and that the hardware and software complement each other seamlessly. This approach not only accelerates the overall development process but also enhances the system's reliability and performance by testing and refining both aspects together.

Through Senior Design 2 we maintained a similar plan to the one discussed above with a few changes. We had multiple hardware issues throughout which caused our software to progress more quickly than the hardware. However, once those major hardware issues were resolved the software and hardware were being developed and tested in parallel once again. Another thing that did deviate from our plan was the enclosure design. Due to the fact that we were trying to

prioritize the portability of our device, it made it a little difficult to allow for ease of access when it came to testing and troubleshooting. The last thing we deviated from was the durability testing. We were able to thoroughly test it in terms of software durability such as upload time and print speed. When it came to physical durability such as a temperature test, we did not execute these.

9.3.1 PCB Assembly

In Senior Design II, one of our top priorities is the prompt assembly of our custom PCB. Completing this step early in the semester is crucial, as it allows us ample time to identify and address any potential design or fabrication issues. By beginning assembly as soon as the boards arrive, we can quickly troubleshoot errors and, if necessary, revise the design and order a corrected version without significantly impacting our overall project timeline. The assembly process will involve carefully soldering each component onto the PCB, following the layout and footprint established during our design phase. This includes placing all surface mount and through-hole components with precision to ensure reliable electrical connections and long-term durability.

A significant part of our initial testing will focus on verifying power delivery across the entire board. We will use multimeters and oscilloscopes to check voltage levels at various points in the circuit, confirming that each component receives the appropriate input without voltage drops or surges. To support this verification process, we are incorporating status LEDs directly into the PCB design. These LEDs will serve as visual indicators to show when specific subsystems, such as microcontrollers, sensors, or communication modules, are powered on. This built-in visual feedback will make it easier to identify issues like disconnected traces, cold solder joints, or failed components. Additionally, it will simplify ongoing diagnostics during both hardware development and field testing.

By prioritizing early PCB assembly and integrating tools for real-time feedback, we aim to streamline the debugging process and ensure a smooth transition from prototyping to final system integration. This methodical approach will help us maintain project momentum and reduce the likelihood of delays due to unforeseen hardware challenges.

In our final design, we implemented a total of five separate PCBs. We began by individually testing each daughterboard before moving on to the main board. Once all boards were verified to function correctly, we revised the main board to correct minor issues identified during initial testing and produced an updated version for final integration.

9.3.2 Software Development

During Senior Design 2, we focused on completing and integrating all essential software components required for our system to function. The software focused on three major areas: mobile application, printer integration, and the firmware running on the ESP32 microcontroller. Each of these components were developed in parallel, with consistent testing to ensure a seamless user experience.

The mobile application serves as an alternative interface to the device's touchscreen, offering users the ability to interact with the system remotely via Bluetooth. It allows users to monitor

weight data transmitted from the device, submit a label print requests, and review past logs of milk storage. The app was designed using the cross-platform framework Flutter to ensure compatibility with both Android and IOS devices. We implemented databases for both the milk log and storage history using Firebase Realtime Database. To support the transmission of data between the ESP32 and mobile application, we defined clear Bluetooth communication protocols serving as an API layer. We considered edge-cases as well to handle events such as data sync conflicts to ensure the app was responsive and accurate. Throughout development, we followed an iterative approach that incorporated user feedback and extensive testing to refine features.

A major focus was integrating all individual software components into a unified and fully functioning system. Initially, different parts of the system such as touchscreen functionality, load cell data processing and label printing are being developed as separate code modules. One of our primary objectives was to merge these subsystems into a cohesive firmware package on the ESP32. This required strategic code structure, shared resource management and redundant testing through each step of the integration process to ensure functionality. Printer integration was a part of the ESP32's firmware, responsible for formatting and sending data to the thermal label printer. We implemented the necessary communication protocols to handle print jobs efficiently, while also managing other potential errors. A large emphasis was placed on ensuring both the touchscreen and mobile app could trigger label printing in a consistent and reliable way. As a part of our final design, we decided to only have label printing triggered through the touchscreen instead of the device itself. This was due to the fact that we had issues with other aspects of the app that were more vital.

The ESP32 firmware is executed to incorporate all hardware interactions and core logic system. This includes reading data from the HX711 load cell amplifier, updating the touchscreen interface, executing print commands, processing user input and managing Bluetooth communication. The firmware will be written in C++ utilizing the Arduino IDE. We decided to actually utilize PlatformIO as opposed to Arduino due to it being recommended as well as it having a faster upload time than the Arduino IDE. It was developed using subsystems and maintainable code structures to support updates and debugging. Additionally, system indicators such as LEDs were controlled through the firmware to provide users with feedback on the system status.

During the mobile application's software testing phase, our primary challenges revolved around three key areas: notification delivery, Bluetooth connectivity, and data synchronization efficiency.

For notifications, we initially underestimated the necessity of integrating multiple libraries for robust functionality. However, parallel development on both Android and iOS platforms facilitated efficient, side-by-side troubleshooting. A platform-specific hurdle later emerged on iOS, requiring a unique ID for notifications to properly appear in the notification center. Bluetooth setup also presented its own set of requirements. We identified missing crucial libraries, such as `flutter_blue_plus`, which proved essential for enabling Bluetooth Low Energy (BLE) compatibility with the ESP32. Additionally, iOS necessitated the implementation of a permission handler to grant the application access to Bluetooth functionalities. Our final Bluetooth obstacle involved the app's initial difficulty in detecting the signal; this was swiftly resolved by leveraging the nRF Radar tool, which ensured reliable device discovery.

Furthermore, we conducted extensive tests for uploading logs from the device. An initial issue arose where device-generated logs appeared separate from manually added entries. This was quickly resolved by implementing a sorting method within the mobile application, a solution validated through multiple trials.

Our last major challenge was optimizing upload times to the mobile application, particularly during our final demo, which relied on a mobile hotspot. The inherent instability of hotspot connections became evident through trials, showing significant variability in upload speed. In contrast, performance over a stable Wi-Fi connection was consistent and reliable.

Despite navigating these technical complexities, our overall software testing process proceeded as smoothly and productively as we could have anticipated.

9.3.3 Hardware Development

For the physical aspect of our project, we need an enclosure that can contain electrical components such as the printer, scale, LEDs, and PCB. In order to do this, we will design a custom 3D-printed enclosure tailored specifically to the layout and dimensions of our hardware. The enclosure will be designed with portability in mind, meaning it must be lightweight, compact, and easy to carry or transport without compromising durability. We will use CAD software such as Fusion 360 or SolidWorks to create a precise digital model, ensuring there is adequate space for each component while maintaining a clean internal layout that supports proper cable management and ventilation. The design will feature specific cutouts for external access to ports, the touchscreen interface, and status indicator LEDs. It will also include a compartment or mounting area to securely house the label printer while allowing printed labels to exit smoothly. Additional features, such as a hinged or removable cover, may be added to facilitate maintenance and battery replacement if needed. Once the design is finalized, the enclosure will be printed using durable yet lightweight materials such as PLA or PETG, which offer sufficient strength and ease of fabrication. After printing, we will test component placement and make any necessary modifications to ensure a snug, secure fit. A well-constructed enclosure is essential not only for protection and organization but also for presenting a polished, professional appearance that enhances user trust and usability in real-world settings.

Chapter 10 Administrative Content

10.1 Budget

The budget provided for this project was \$500. This funding was supplied by our team through our sponsor, and our goal throughout development was to remain as cost-efficient as possible while meeting all the objectives for our prototype.

Table 20: Estimated Pumped Project Budget

Product Name	Cost
Microcontroller	\$10-\$35
Load-Cell	\$6-\$10
Touchscreen Display	\$20-\$60

Thermal Label Printer	\$30-\$120
Thermal Label Paper	\$10-\$30
Power Supply	\$20
Wires	\$40
PCBs + Tariffs	\$80-\$150
LEDs	\$5-\$10
Load-Cell Amplifier	\$40-\$50
Rechargeable battery	\$9-\$20
3D Filament	\$40
Battery Charging IC/module	\$5-\$20
Voltage Regulator	\$1-\$5
Prototyping	\$100-\$150

10.2 Bill of Materials

The Bill of Materials (BOM) served as a comprehensive inventory of every component and material required for the successful construction and implementation of our project. It was a critical document that enabled us to manage resources efficiently, maintain a clear overview of our hardware dependencies, and track costs throughout the development process. Each item listed in the BOM was specified by its component name, quantity needed, manufacturer, part number (if applicable), and price. This level of detail ensured that procurement was streamlined and that there was no ambiguity when sourcing parts. Additionally, we included multiple entries for similar components, such as different versions of microcontrollers or thermal printers, to provide flexibility in case certain parts needed substitution.

Table 21: Bill of Materials

Component	Unit(s)	Manufacturer	Manufacturer Part number	Price
3.5 Inches TFT LCD Touch Screen	1	Hosyond	ILI9488	\$16.99
Microcontroller	3	Espressif Systems	ESP32-DEVKITC-32UE	\$24.27
Microcontroller	3	AYWHP	ESP32-WROOM-32	\$15.99
Load-Cell	2	SparkFun Electronics	SEN-13329	\$19.38

3.3 V Buck Regulator	4	Texas Instruments	LM2576SX-3.3/NOPB	\$14.88
9V Boost Regulator	4	Texas Instruments	LM3478MM/NOPB	\$11.00
IC Transceiver	3	Texas Instruments	MAX3232ECPWR	\$2.82
Buck Regulator	4	Texas Instruments	TPS561208DDCR	\$2.08
Micro Thermal Printer	1	PRIMUZ	MC208W	\$48.98
Embedded Thermal Printer	1	DFROBOT	DFR0503-EN	\$39.00
Thermal Label Paper	3	PHOMEMO	Not specified	\$6.99
PCB Design	10	JLPCB	Not Specified	\$43.77
LED(blue) 0805 SMD	8	Würth Elektronik	150080BS75000	\$1.36
LED(green) 0805 SMD	8	Würth Elektronik	150060VS75000	\$1.36
LED(red) 0805 SMD	8	Würth Elektronik	150080SS7500	\$1.36
Loadcell-Amplifier HX711	2	SparkFun Electronics	SEN-13879	\$21.90
Digital Load Cell Weight Sensor 5Kg	2	China IC Sales Store	Not Specified	\$8.40
Digital Load Cell Weight Sensor 10kg	1	China IC Sales Store	Not Specified	\$4.20
Acrylic Blanks	6	Hobby Lobby	2385318	\$2.99
Screws	32	Everbilt	1006547890	\$9.01
Spacers	2	Home Depot	Not specified	\$4.50
Solderless Breadboards	4	ELEGOO	Not specified	\$6.99
18650 Li-ion Battery	10	18650batterystore	Not specified	\$35.00
IC BATT CHG LI-ION 1CELL 10MSOP	4	Microchip Technology	MCP73837-NVI/UN	\$6.04
Digital Load Cell Weight Sensor 5KG Module		Aliexpress	Not Specified	\$20.33

Alitove 12V Power Supply Adapter	1	Amazon	Not Specified	\$10.65
PCB Components		Mouser, Digikey	Thin film resistors, capacitors, LEDs, Soder Stencils, IC Bridge, Diodes,	\$307.63
Talent Cell Rechargeable 12 Battery	1	Amazon	Not Specified	\$35.13
BreastMilk Storage Bags	1	Amazon	Not Specified	\$6.39
Primuz Thermal Printer	1	Amazon	Not Specified	\$49.93
Wires		Amazon	2.0 Cable Pigtail Extension, DC Power 3FT DC 5.5MM x 2.1MM Female, Jumper Cables	\$30.94
3D Filament		Amazon, 3D Print Orlando	ELEGOO PLA	\$64.02

10.3 Distribution of Work

Our goal was to distribute our work efficiently, ensuring each team member could focus on their strengths and contribute to the project in a meaningful way. By breaking down our tasks into manageable portions, we remained organized and made consistent progress toward our end goal. This structured approach not only promoted productivity but also allowed us to leverage each member's expertise to optimize the overall quality and functionality of our final product.

The electrical engineering members of the team were primarily responsible for the design, testing, and implementation of the hardware components. Their tasks included developing the custom PCB layout, selecting appropriate electrical components, ensuring proper power distribution, and validating signal integrity across the system. They were also in charge of integrating the load cell and HX711 module with the microcontroller and ensuring all devices—such as the touchscreen, LEDs, and label printer—were properly connected and powered. Their role was critical in ensuring that the system was not only functional but also electrically safe and efficient.

Meanwhile, the computer engineering members focused on the development of the software that brought the system to life. This included programming the microcontroller, creating the user interface on the touchscreen, implementing communication protocols for the label printer, and developing backend logic such as inventory tracking and expiration alerts. They also worked on

integrating wireless communication features like Bluetooth and ensured the reliability of data transmission between the device and the mobile application. Additionally, the computer engineers were responsible for debugging, testing software modules, and documenting the system's operation for future updates or troubleshooting.

Regular communication and collaboration between both disciplines were essential in ensuring the hardware and software were well-aligned and operated seamlessly together. By dividing responsibilities according to our areas of expertise and maintaining a shared vision, we successfully created a cohesive, functional, and polished final product that met both technical and user experience goals.

Table 22: Distribution of Work

Requirements	Responsible Team Member(s)
OnShape CAD Design	Johan
Mobile Application Design Development	Alexandra, Mya
Device Software Design and Implementation	Mya, Alexandra
Printed Circuit Board Design	Simran, Johan
Weight Sensor Design and Implementation	Johan, Simran
Mobile Application Debugging	Mya, Alexandra
Battery Intergration and Management	Simran
LED Control System	Simran, Mya
Display Intergration	Alexandra
Wireless Communication	Alexandra, Mya
Thermal and Durability Testing	Simran, Johan
Testing and Debugging	All

10.4 Milestones

In Senior Design I, our team operated within a 16-week project timeline focused on research, planning, and documentation for prototype development. This initial phase of the senior design sequence was critical, as it laid the groundwork for the successful implementation and testing of our system in the following semester. Throughout this period, we adopted a structured and iterative project management approach that emphasized continuous improvement and thorough documentation.

Each week, we updated our documentation to reflect the latest progress in hardware and software development. These updates included revisions to our system architecture diagrams, schematic designs, software flowcharts, and component selection justifications. Maintaining current documentation allowed us to identify potential integration issues early and resolve design challenges proactively.

In parallel with technical development, we held regular internal meetings and maintained ongoing communication with our sponsors, who played a key role in guiding design decisions and ensuring our deliverables aligned with the intended application. These discussions served as checkpoints for feedback, clarification of requirements, and alignment with the sponsor's vision. We documented these interactions in detail, including action items, feedback summaries, and evolving requirements, which were incorporated into our project records.

We also followed a structured timeline with well-defined milestones such as completing the project proposal, developing the system block diagram, validating component selections, conducting design reviews, and submitting the final written report. Each milestone was planned to build on the previous one, resulting in a cohesive and comprehensive design by the end of the semester. By adhering to this organized plan and emphasizing communication, iteration, and sponsor engagement, we established a strong foundation for the prototyping and implementation work carried out in Senior Design II.

Table 23: Pumped Project Timeline

Weeks	Important Tasks + Due Dates
1 (Jan. 5 th – Jan. 11 th)	Form Group and Discuss Ideas
2 (Jan. 12 th – Jan. 18 th)	Finalize Idea and begin Divide & Conquer Document
3 (Jan. 19 th – Jan. 25 th)	Meet with Sponsor, Divide & Conquer Document Due
4 (Jan. 26 th – Feb. 1 st)	Divide & Conquer Meeting and Revisions Research the software needed to create the app.
6-7 (Feb. 9 th – Feb. 22 nd)	Research scale and label printer designs
8 (Feb. 23 rd – Mar. 1 st)	Start PCB design
9-10 (Mar. 2 nd – Mar. 15 th)	Continue software and PCB development
11 (Mar. 16 th – Mar. 22 nd)	Continue software and PCB development
12 (Mar. 23 rd – Mar. 29 th)	Midterm Report and Meeting
13 (Mar. 30 th – Apr. 5 th)	Midterm Report Revisions and Upload
14-15 (Apr. 6 th – Apr. 19 th)	Continue Report Revisions
16 (Apr. 20 th – Apr. 22 nd)	Final Documentation and Demo
17 (Apr. 23 rd – May 9 th)	Finalize first PCB design and continue app development
18 (May 12 th – May 17 th)	Get feedback on PCB, make necessary revisions and order PCB

19(May 18 th – May 25 th)	Continue developing mobile application and embedded software
20(May 26 th – May 30 th)	Continue developing mobile application and embedded software Make a plan for hardware assembly
21(May 31 st – June 7 th)	Receive PCB and begin component soldering Continue App integration
22(June 8 th – June 14 th)	Debug PCB issues and if possible start integrating software with hardware
23(June 15 th – June 21 st)	Continue integration, conduct functional testing and collect test data Begin enclosure design and 3D printing
24-25(June 22 nd – July 5 th)	Optimize software for Bluetooth/ Data syncing Conduct full system testing Debug software + hardware issues Document design changes
26(July 6 th – July 12 th)	Prepare presentation materials Record demo videos Update diagrams and schematics
27(July 13 th – July 19 th)	Final testing and user experience validation Polish UI and improve system responsiveness
28(July 20 th – July 29 th)	Final report submission Final project demo + presentation

Chapter 11 Conclusion

With our project, our aim is to create a product to help alleviate the stress of parents – specifically mothers. As we learned from our project sponsors Victoria Rodriguez and Matthew Aberman, the process of pumping breastmilk can occur at very early or very late times in the day where parents may be prone to errors in labelling. This device, designed with both practicality and compassion in mind, embodies our commitment to engineering innovation that impacts lives in ways that may go unseen by most. We hope our product can streamline the milk labeling and storage process, making it easier to manage and track bags of breastmilk. By reducing stress and ensuring accurate storage, we hope to improve the health and well-being of both mother and child. To achieve this, we will create a physical component containing a scale and printer along with a microcontroller capable of communicating with a mobile device, where our developed software application will keep inventory of the data from the scale.

In Senior Design I, we dedicated significant time and effort to the foundational stages of our project, focusing heavily on planning, research, design, and prototyping. We began by establishing a comprehensive set of goals, ranging from essential functions to more advanced features, that served as the guiding framework for our design decisions. Among the core objectives were the ability to accurately measure the volume of pumped breastmilk, generate printed labels that include key information such as date, time, and volume, and provide a reliable digital interface through our mobile application for inventory tracking and management. These

goals were not only technically driven but also informed by the real-world needs of our target users, which we identified through sponsor interviews and user scenarios. By clearly defining our desired outcomes early in the process, we were able to make informed hardware and software choices that aligned with our vision for a functional, user-friendly, and impactful final product.

After gaining inspiration from similar existing products and projects, we began the process with research into our different hardware sections - namely the scale, the printer, and the display - as well as the other components needed to execute them. We explored and evaluated different components including microcontrollers, power regulators, display modules, and thermal printers. Upon identifying each of the electrical components we needed, we compared different models of components in order to find the best options for the needs of our project. This thorough investigation granted us the flexibility to reconsider options and explore alternatives down the line, but we ultimately decided on specific parts to proceed with the development and testing of our hardware. We selected the ESP32-WROOM-32 microcontroller for its WiFi and Bluetooth capabilities which will allow us to integrate our product with mobile app connectivity. For the display, we went with a 4.3" TFT IPS LCD screen that will provide clear visual feedback and be navigable by touch. We considered both buck and boost regulators for stable power delivery across our components, but we ended up only using buck regulators.

For the software component, we laid a strong foundation for the mobile application by drafting an intuitive and accessible user interface that aligns with the daily routines of breastfeeding parents. Through the use of detailed software flowcharts and system architecture mapping, we strategically defined how data such as milk volume, time of pumping, and expiration dates would flow through the application. This planning stage was essential in establishing a reliable and scalable structure for both front-end interaction and back-end data handling. We also identified and integrated essential libraries and frameworks that will facilitate Bluetooth communication, data storage, and dynamic label formatting, ensuring seamless interaction between the app and the device hardware. Moreover, we began structuring the microcontroller firmware to handle sensor inputs, command interpretation, and real-time feedback, bridging the gap between physical device functionality and digital interface responsiveness.

Outside of the technical build, we considered the real-world implications of our design by researching applicable standards and constraints that a product like ours would need to comply with. Notable standards include those related to electrical safety, power efficiency, and user data protection, particularly because our app requires login functionality and may store sensitive information. This process was assisted greatly by AI, and thus another section of this report is dedicated to gaining a deeper understanding of AI and analyzing our use of it for the project.

The bulk of the progress we've made in Senior Design I has been in our research and hardware development. After selecting our components, we ordered the necessary parts and began assembling them on a breadboard to test for functionality. This proved to be a challenging process, as there were several times where we encountered inaccurate results or no results at all. However, these setbacks became valuable learning opportunities that helped us refine our initial prototypes and understand what did and didn't work with our system, which will be useful going forward as we continue to integrate all parts of our project together. Eventually, we were able to

provide basic demonstrations of functionality of our scale, printer, and display screen, signifying key progress in both our hardware and software development.

As a team with limited prior project experience, much of our process was trial and error as we encountered design problems for the first time. Although technical information was accessible to us from online resources and guidance from mentors and professors, one major problem we ran into was time management and understanding the course timeline. Our biggest setback occurred around the middle of the semester when our midterm report was deemed severely lacking in content, placing us at risk of failure in the course. As a group, we quickly remedied this by devoting the following week to increasing our research, creating electrical schematics and software flowcharts. We were ultimately able to get back on track with the timeline of the course, and the pressure helped mark a turning point in staying on top of the workload required for the project.

By the end of Senior Design II, we successfully completed the remaining critical components necessary to realize our final product. We finalized and refined our PCB designs, ensuring efficient and clean board layouts across all five boards. Our mobile app development was completed, including the implementation of a structured database and a fully functional user account management system. We also designed, modeled, and 3D printed a custom enclosure to securely house all hardware components. The printer, a key aspect of our system, was successfully integrated using a method that balanced complexity and project scope. To ensure system reliability, we conducted extensive stress testing and performance analysis, confirming the device's durability, accuracy, and overall stability. Ultimately, we delivered a fully functional, portable, and user-friendly breastmilk labeling and tracking device. The experience we gained throughout both phases of Senior Design greatly enhanced our engineering capabilities and allowed us to bring our initial vision to life.

Appendix A: References

- [1] "AI: Artificial Intelligence - What Is the Definition of AI and How Does AI Work?" University of Illinois Chicago College of Engineering, <https://meng.uic.edu/news-stories/ai-artificial-intelligence-what-is-the-definition-of-ai-and-how-does-ai-work/>. Accessed 22 Apr. 2025.
- [2] "Artificial Intelligence (AI)." Built In, <https://builtin.com/artificial-intelligence>. Accessed 22 Apr. 2025.
- [3] "ChatGPT vs. Microsoft Copilot vs. Gemini: Which Is the Best AI Chatbot?" ZDNet, <https://www.zdnet.com/article/chatgpt-vs-microsoft-copilot-vs-gemini-which-is-the-best-ai-chatbot/>. Accessed 22 Apr. 2025.
- [4] "Firebase Realtime Database." GeeksforGeeks, <https://www.geeksforgeeks.org/firebase-realtime-database/>. Accessed 7 Mar. 2025.
- [5] "Getting Started with Cross-Platform Mobile Application Using Flutter." GeeksforGeeks, <https://www.geeksforgeeks.org/getting-started-with-cross-platform-mobile-application-using-flutter/>. Accessed 10 Mar. 2025.
- [6] "Google Bard vs ChatGPT vs Copilot." GeeksforGeeks, <https://www.geeksforgeeks.org/google-bard-vs-chatgpt-vs-copilot/>. Accessed 22 Apr. 2025.
- [7] "How Does AI Actually Work?" Colorado State University Global, <https://csuglobal.edu/blog/how-does-ai-actually-work>. Accessed 22 Apr. 2025.
- [8] "How Does AI Work?" Coursera, <https://www.coursera.org/articles/how-does-ai-work>. Accessed 22 Apr. 2025.
- [9] "How to Use MAX3232 RS232 to UART Converter with ESP32." Electronics Stack Exchange, 22 Feb. 2023, <https://electronics.stackexchange.com/questions/624029/how-to-use-max3232-rs232-to-uart-converter-with-esp32>. Accessed 21 Apr. 2025.
- [10] "Machine Learning." IBM, <https://www.ibm.com/think/topics/machine-learning>. Accessed 22 Apr. 2025.
- [11] 3DPrinting.com. 3D Printing Software. [Online]. Available: <https://3dprinting.com/software/>. Accessed: 12 Mar. 2025
- [12] ASP, Inc. ASP 53037 Material Safety Data Sheet (MSDS). [Online]. Available: <https://files.batteryjunction.com/frontend/files/asp/msds/ASP-53037-MSDS.pdf>. Accessed: 2 Mar. 2025.
- [13] Apple Inc. Swift. Apple Developer, <https://developer.apple.com/swift/>. Accessed 10 Mar. 2025.
- [14] Avnet. Comparing Common MCU Interface Protocols. [Online]. Available: <https://www.avnet.com/wps/portal/us/resources/article/comparing-common-mcu-interface-protocols/>. Accessed: 20-Feb-2025
- [15] Barcodes Inc. Zebra Breastmilk Tracking Solutions. Barcodes Inc., <https://www.barcodesinc.com/zebra/breastmilk-tracking.htm>. Accessed 2 Feb. 2025.
- [16] Batteries, Inc. Primary vs. Secondary Batteries. [Online]. Available: <https://batteriesinc.net/primary-vs-secondary-batteries/>. Accessed: 2-Mar-2025
- [17] Bitbucket: Key Terms to Know. Bitbucket, <https://bitbucket.org/product/guides/getting-started/overview#key-terms-to-know>. Accessed 15 Mar. 2025.
- [18] Bluetooth SIG. Bluetooth Specifications. [Online]. Available: <https://www.bluetooth.com/specifications/specs/>. Accessed: 10-Mar-2025
- [19] Centers for Disease Control and Prevention. Preparation of Breast Milk. https://www.cdc.gov/breastfeeding/pdf/preparation-of-breast-milk_h.pdf. Accessed 21 Apr. 2025.
- [20] Codecademy. "The History of Chatbots." Codecademy, <https://www.codecademy.com/article/history-of-chatbots> Accessed 13 Mar. 2025.

- [21] Constellation. Solar Energy: Pros and Cons. [Online]. Available: <https://www.constellation.com/energy-101/energy-innovation/solar-energy-pros-and-cons.html>. Accessed: 2-Mar-2025
- [22] Consumer Product Safety Commission (CPSC). Battery Regulations and Standards. [Online]. Available: <https://www.cpsc.gov/Regulations-Laws--Standards/Voluntary-Standards/Topics/Batteries>. Accessed: 2-Mar-2025
- [23] Cubtale. Smart Milk Inventory Tracking Stickers. Cubtale, <https://www.cubtale.com/products/smart-milk-inventory-tracking-stickers>. Accessed 4 Feb. 2025.
- [24] DNV. ISO 22000 - Food Safety Management. DNV, <https://www.dnv.com/services/iso-22000-food-safety-management-5173/#:~:text=The%20ISO%2022000%20%2D%20food%20safety,as%20statutory%20and%20regulatory%20demands>. Accessed 13 Mar. 2025.
- [25] EEMB. ASR00050 18650 2500mAh Datasheet. [Online]. Available: https://www.mouser.com/datasheet/2/855/ASR00050_18650_2500mAh-3078640.pdf. Accessed: 2-Mar-2025
- [26] Eaton. Supercapacitors vs. Batteries. [Online]. Available: <https://www.eaton.com/us/en-us/products/electronic-components/topics/supercapacitors-vs-batteries.html>. Accessed: 2-Mar-2025.
- [27] Electronics For You. E-Paper Display: Advantages & Disadvantages. [Online]. Available: <https://www.electronicsforu.com/technology-trends/learn-electronics/e-paper-display-advantages-disadvantages>. Accessed: 4-Mar-2025.
- [28] Electronics For You. LCD (Liquid Crystal Display) Basics. [Online]. Available: <https://www.electronicsforu.com/technology-trends/learn-electronics/lcd-liquid-crystal-display-basics>. Accessed: 4-Mar-2025
- [29] Espressif Systems. ESP32-WROOM-32E/ESP32-WROOM-32UE Datasheet. 17 May 2021, https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32e_esp32-wroom-32ue_datasheet_en.pdf. Accessed 21 Apr. 2025.
- [30] GitHub Definition. TechTarget, <https://www.techtarget.com/searchitoperations/definition/GitHub>. Accessed 15 Mar. 2025.
- [31] Hot Reload. Flutter Documentation, <https://docs.flutter.dev/tools/hot-reload>. Accessed 7 Mar. 2025.
- [32] Huawei. HarmonyOS. [Online]. Available: <https://www.harmonyos.com/en/>. Accessed: 10-Mar-2025
- [33] IBM. The History of Artificial Intelligence. IBM, <https://www.ibm.com/think/topics/history-of-artificial-intelligence>. Accessed 13 Mar. 2025.
- [34] IEEE. IEEE 802.15.1 Standard. [Online]. Available: <https://standards.ieee.org/ieee/802.15.1/1180/>. Accessed: 10-Mar-2025
- [35] Instructables. "How to Build Arduino Weighing Scales." Instructables, www.instructables.com/How-to-Build-Arduino-Weighing-Scales/. Accessed 19 Jan. 2025.
- [36] International Electrotechnical Commission (IEC). IEC Publication 62790. [Online]. Available: <https://webstore.iec.ch/en/publication/62790>. Accessed: 10-Mar-2025
- [37] International Organization for Standardization (ISO). ISO/IEC 27001 – Information Security Management. [Online]. Available: <https://www.iso.org/standard/27001>. Accessed: 10 Mar. 2025
- [38] Introduction to MySQL. Hostinger, <https://www.hostinger.com/tutorials/what-is-mysql>. Accessed 8 Mar. 2025.
- [39] Klarus. KLARUS 16340 700mAh Datasheet. [Online]. Available: <https://www.shorepowerinc.com/media/wysiwyg/files/klarus/datasheet/KLARUS-16340-700-DataSheet.pdf>. Accessed: 2 Mar. 2025.

- [40] LCDWiki. 3.5inch SPI Module MSP3520 User Manual.
http://www.lcdwiki.com/res/MSP3520/3.5inch_SPI_Module_MSP3520_User_Manual_EN.pdf.
Accessed 21 Apr. 2025.
- [41] MicroPython. MicroPython, <https://micropython.org/>. Accessed 11 Mar. 2025.
- [42] Microchip Technology Inc. AN1149: Interfacing UART Based Serial Devices to PIC® Microcontrollers. July 2008, <https://ww1.microchip.com/downloads/en/AppNotes/01149c.pdf>.
Accessed 21 Apr. 2025.
- [43] Newhaven Display. NHD-4.3-480272FT-CSXP-T Specification. [Online]. Available:
<https://newhavendisplay.com/content/specs/NHD-4.3-480272FT-CSXP-T.pdf>. Accessed: 10-Mar-2025
- [44] PCBONLINE. Top 6 Popular Microcontrollers. [Online]. Available:
<https://www.pcbonline.com/blog/top-6-popular-microcontrollers.html>. Accessed: 12-Mar-2025
- [45] React Native Definition. Netguru, <https://www.netguru.com/glossary/react-native>.
Accessed 10 Mar. 2025.
- [46] Real-time Database Documentation. Firebase, <https://firebase.google.com/docs/database>.
Accessed 7 Mar. 2025.
- [47] Riverdi. OLED vs. LCD: A Comprehensive Comparison. [Online]. Available:
<https://riverdi.com/blog/oled-vs-lcd-a-comprehensive-comparison>. Accessed: 4-Mar-2025.
- [48] Rui, Rui. "ESP32 with Load Cell and HX711 – Build a Digital Scale." Random Nerd Tutorials, <https://randomnerdtutorials.com/esp32-load-cell-hx711/#digital-scale-esp32>. Accessed 21 Apr. 2025.
- [49] SEACOMP. 5-Inch TFT LCD DT050BTFT Specification. [Online]. Available:
<https://www.seacomp.com/product/5-inch-tft-lcd-dt050btft>. Accessed: 10-Mar-2025
- [50] SEACOMP. DT043CTFT-IPS Display Specification. [Online]. Available:
<https://www.seacomp.com/sites/default/files/datasheets/DT043CTFT-IPS-Displaytech-Spec.pdf>.
Accessed: 10-Mar-2025
- [51] SQLite About Page. SQLite, <https://www.sqlite.org/about.html>. Accessed 7 Mar. 2025.
- [52] ScienceDirect. "Cross-platform Mobile Application Development: A Review of Features and Challenges." ScienceDirect,
<https://www.sciencedirect.com/science/article/abs/pii/S0950584920301439>. Accessed 9 Mar. 2025.
- [53] SparkFun Electronics. SparkFun HX711 Load Cell Amplifier Hookup Guide v1.1.
https://cdn.sparkfun.com/datasheets/Sensors/ForceFlex/SparkFun_HX711_Load_Cell_v1.1.pdf.
Accessed 21 Apr. 2025.
- [54] SparkFun Electronics. SparkFun Load Cell - 5kg (SEN-13329). SparkFun Electronics, 2025, www.sparkfun.com/products/13329.
- [55] Swift. Swift.org, <https://www.swift.org/>. Accessed 10 Mar. 2025.
- [56] Tech Explaining. "How to Use HX711 Load Cell Amplifier with ESP32 | DIY Digital Scale." YouTube, 3 Jan. 2023, <https://www.youtube.com/watch?v=NidjZOxMMxo>. Accessed 21 Apr. 2025.
- [57] Tech Explorations. "ESP32 Digital Output: How to Control Outputs."
<https://techexplorations.com/guides/esp32/begin/digitalout/>. Accessed 21 Apr. 2025.
- [58] Texas Instruments. MAX3232 Datasheet. Mar. 2015,
<https://www.ti.com/lit/ds/symlink/max3232.pdf>. Accessed 21 Apr. 2025.
- [59] Texas Instruments. Weigh Scale Design Using Load Cell and ADS1232 (TIDUAC1A). Texas Instruments, 2015, www.ti.com/lit/ug/tiduac1a/tiduac1a.pdf.
- [60] Tiny Circuits. ASR00050 18650 2500mAh Datasheet. [Online]. Available:
https://www.mouser.com/datasheet/2/855/ASR00050_18650_2500mAh-3078640.pdf. Accessed: 2-Mar-2025.

- [61] US Micro Products. Introduction to OLED Technology & Its Pros/Cons. [Online]. Available: <https://www.usmicroproducts.com/blog/introduction-oled-technology-its-proscons>. Accessed: 4 Mar. 2025.
- [62] United Nations Environment Programme. "AI Has an Environmental Problem. Here's What the World Can Do About It." UNEP, <https://www.unep.org/news-and-stories/story/ai-has-environmental-problem-heres-what-world-can-do-about>. Accessed 13 Mar. 2025.
- [63] United States Food and Drug Administration. Food Packaging & Other Substances that Come into Contact with Food - Information for Consumers. FDA, <https://www.fda.gov/food/food-ingredients-packaging/food-packaging-other-substances-come-contact-food-information-consumers>. Accessed 13 Mar. 2025.
- [64] Upesy Electronics. "ESP32 Pinout Reference: GPIO Pins Ultimate Guide." <https://www.upesy.com/blogs/tutorials/esp32-pinout-reference-gpio-pins-ultimate-guide#>. Accessed 21 Apr. 2025.
- [65] Uswitch. Mobile Operating Systems Guide. [Online]. Available: <https://www.uswitch.com/mobiles/guides/mobile-operating-systems/>. Accessed: 10 Mar. 2025
- [66] W3Schools. "C Language Introduction." W3Schools, https://www.w3schools.com/c/c_intro.php. Accessed 10 Mar. 2025.
- [67] Zebra Technologies. "Mobile Printer Battery FAQ." Zebra Technologies, www.zebra.com/content/dam/zebra_dam/en/faq/mobile-printer-battery-faq-a4-en-us.pdf. Accessed 19 Jan. 2025.

Declaration: We hereby declare that we have not copied more than 7 pages from the Large Language Model (LLM). We have utilized LLM for drafting, outlining, comparing, summarizing, and proofreading purposes.

Appendix D: Software Code

Initial Code used to calibrate and test the load cell:

```
#include "HX711.h"
// HX711 circuit wiring
const int LOADCELL_DOUT_PIN = 27;
const int LOADCELL_SCK_PIN = 26;
HX711 scale;
void setup() {
  Serial.begin(9600);
  scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);
}
void loop() {
  if (scale.is_ready()) {
    long reading = scale.read();
    Serial.print("HX711 reading: ");
    Serial.println(reading);
  } else {
    Serial.println("HX711 not found.");
  }
  delay(1000);
}
```

Code used in demo for homepage and weight reading:

```
#include <TFT_eSPI.h>
#include <SPI.h>
#include "HX711.h"
// Load Cell pins
const int LOADCELL_DOUT_PIN = 27;
const int LOADCELL_SCK_PIN = 26;
float calibration_factor = -215.64;
HX711 scale;
// TFT display
TFT_eSPI tft = TFT_eSPI();
// Coordinates where the weight will be shown
const int weightTextX = 80;
const int weightTextY = 100;
void setup() {
  Serial.begin(9600);
  // Initialize scale
  scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);
  scale.set_scale(calibration_factor);
  scale.tare();
  Serial.println("HX711 initialized and tared.");
  // Initialize screen
  tft.init();
```

```

tft.setRotation(1);
tft.fillScreen(TFT_WHITE);
// Date and Time
tft.setTextSize(2);
tft.setTextColor(TFT_BLACK, TFT_WHITE);
tft.setCursor(10, 10);
tft.print("April 4, 2025");
tft.setCursor(240, 10);
tft.print("10:32 AM");
// Weight Box
tft.fillRect(10, 40, 300, 160, TFT_SKYBLUE);
tft.setTextSize(4);
tft.setTextColor(TFT_BLACK, TFT_SKYBLUE);
tft.setCursor(weightTextX, weightTextY);
tft.print("0.00 oz");
// Sidebar and Bottom Buttons
drawButton(320, 40, 100, 40, "Mode");
drawButton(320, 90, 100, 40, "Settings");
drawButton(320, 140, 100, 40, "Restart");
drawButton(10, 210, 100, 40, "Print");
drawButton(120, 210, 100, 40, "Back");
drawButton(230, 210, 100, 40, "Sleep Mode");
}
void drawButton(int x, int y, int w, int h, const char* label) {
  tft.drawRect(x, y, w, h, TFT_BLACK);
  tft.setTextColor(TFT_BLACK);
  tft.setTextSize(2);
  int textX = x + (w - strlen(label) * 12) / 2;
  int textY = y + (h - 16) / 2;
  tft.setCursor(textX, textY);
  tft.print(label);
}
void loop() {
  if (scale.is_ready()) {
    float weight_grams = scale.get_units();
    float weight_oz = weight_grams / 28.3495;
    // Print to serial monitor
    Serial.print("Weight: ");
    Serial.print(weight_oz, 2);
    Serial.println(" oz");
    // Clear previous weight text area
    tft.fillRect(weightTextX, weightTextY, 180, 40, TFT_SKYBLUE);
    // Display updated weight
    tft.setTextSize(4);

```

```
tft.setTextColor(TFT_BLACK, TFT_SKYBLUE);  
tft.setCursor(weightTextX, weightTextY);  
tft.printf("%.2f oz", weight_oz);  
} else {  
  Serial.println("HX711 not found.");  
}  
delay(1000);  
}
```

Appendix E: ChatGPT Prompts & Outcomes

E.1: “what would be the best lithium-ion batteries to use for the project”

ChatGPT said:

The best **lithium-ion battery** for your project depends on factors like power consumption, size, weight, and rechargeability. Since your project involves **LEDs as a guideline on a device**, I assume you need a **compact, rechargeable, and efficient power source**. Here are some good lithium-ion battery options:

Best Lithium-Ion Battery Options for Your Device:

1. 18650 Li-ion Battery (3.7V, ~2200mAh - 3500mAh)

✓ **Best for:** Mid-to-high power consumption devices that need good battery life.

🔋 **Capacity:** 2200mAh - 3500mAh per cell.

⚡ **Voltage:** 3.7V nominal (4.2V fully charged).

📏 **Size:** 18mm diameter, 65mm length (cylindrical).

🔄 **Rechargeable?** Yes, supports many charge cycles (~500+).

Why? Commonly used in flashlights, power banks, and small electronics. A **single** 18650 cell may be enough, or you can use multiple in **series or parallel** to match your voltage/current needs.

2. LiPo (Lithium Polymer) Battery Pack (3.7V - 7.4V, 500mAh - 5000mAh)

✓ **Best for:** Compact, lightweight, and custom-shaped designs.

🔋 **Capacity:** 500mAh - 5000mAh (varies based on size).

⚡ **Voltage:** Usually 3.7V per cell (or 7.4V for a 2S pack).

📏 **Size:** Slim and flat (varies by capacity).

🔄 **Rechargeable?** Yes, but more delicate than cylindrical Li-ion cells.

Why? If space is a concern, **LiPo batteries** are flexible and can be shaped to fit into a custom enclosure. However, they require **careful charging** to avoid swelling.

3. 14500 Li-ion Battery (3.7V, ~800mAh - 1200mAh)

✓ **Best for:** Compact, AA-sized battery-powered devices.

🔋 **Capacity:** 800mAh - 1200mAh.

⚡ **Voltage:** 3.7V per cell.

📏 **Size:** Same as AA batteries (14mm diameter, 50mm length).

🔄 **Rechargeable?** Yes.

Why? If your device is small and currently designed for **AA batteries**, this is a great drop-in **rechargeable** replacement (but be mindful of the voltage difference).

4. Lithium-ion Battery Packs (7.4V - 12V, 2000mAh - 10,000mAh)

✓ **Best for:** Higher power requirements, longer run time.

🔋 **Capacity:** 2000mAh - 10,000mAh.

⚡ **Voltage:** 7.4V (2-cell), 11.1V (3-cell), 12V (multi-cell packs).

📏 **Size:** Varies, but generally larger.

🔄 **Rechargeable?** Yes, great for continuous use.

Why? If your device needs more **power or longer battery life**, a **multi-cell Li-ion pack** could work. These are commonly used in **robots, LED strips, and portable tools**.

E.2: “What are some standards and constraints that should be considered for this product?”

ChatGPT said:



STANDARDS TO CONSIDER

1. Health and Safety Standards (for food-contact materials)

FDA Food Contact Regulations (21 CFR Part 177)

Any part of your device that comes in direct contact with breastmilk (or storage bags) must be made from **FDA-compliant food-grade materials**.

→ Materials should be **BPA-free, non-toxic, and easy to sanitize**.

ISO 22000 (Food Safety Management)

If you ever plan to expand this device into commercial/clinical use, aligning with **ISO food safety** practices is a big plus. Covers hygiene, contamination prevention, and safe handling.

2. Labeling and Traceability Standards

Human Milk Banking Association of North America (HMBANA) Guidelines

If your device supports milk donation, you should look at HMBANA standards for **labeling breastmilk**:

Date and time of expression.

Donor information.

Milk treatment (if pasteurized).

Storage conditions.

Labels should be **clear, legible, and durable** in freezing/thawing conditions.

GS1 Standards (Optional for future-proofing)

If you envision this system scaling to clinics or milk banks, **GS1 barcodes or QR codes** are used for healthcare traceability. They can hold unique donor or batch data for better **inventory management**.

3. Electrical and Electronic Safety Standards

UL/IEC 62368-1 (Audio/Video, IT and Communication Technology Equipment Safety)

If the device is **battery-operated** and has **electronic components**, you may need to ensure compliance with general electronic safety standards.

→ Focus on **safe voltage levels, overheating, and battery safety**.

FCC Part 15

If your device uses **Bluetooth or Wi-Fi**, it must comply with **FCC standards for electromagnetic interference (EMI)** to ensure it doesn't disrupt other devices.

4. Battery and Charging Standards

IEC 62133-2 (Rechargeable Battery Safety)

For **portable, battery-powered devices**, especially lithium-ion batteries, this standard addresses **thermal runaway, short circuit, overcharge protection**, etc.

5. Software Standards

ISO/IEC 25010 (Software Quality)

Ensures your app/software design focuses on **usability, reliability, and security**.

HIPAA (Health Insurance Portability and Accountability Act)

If you store **personal health information** (baby feeding data, donor details), you should **encrypt** user data and comply with **privacy and security** requirements.



DESIGN CONSTRAINTS TO CONSIDER

1. Portability

The device should be **lightweight, compact, and easy to transport**, since moms may need it at home, work, or while traveling.

Battery-powered operation is key, limiting **power consumption** and requiring **efficient energy use**.

2. Ease of Use

One-handed operation (moms may be holding a baby).

Simple user interface (large buttons or touch areas, intuitive design).

Fast and **hassle-free setup**—no complicated pairing or calibration each use.

3. Accuracy and Precision

Scale must have **high sensitivity** to measure small volumes (typically 1-10 oz), ideally accurate to **±1 gram or better**.

Measurement needs to **account for tare weight** (the bag/container).

4. Environmental Constraints

Must be **resistant to spills, moisture, and condensation** (since milk can leak).

Operate at **common indoor temperatures**, and components should tolerate **freezer** or **refrigerated** conditions if needed.

Label printer should produce **smudge-proof, water-resistant** labels that adhere even in **cold or damp conditions**.

5. Cost Constraints

Affordable for end-users (mothers, milk banks, healthcare providers).

Use **off-the-shelf components** when possible to minimize **development and manufacturing costs**.

6. Maintenance and Durability

Easy to **clean** and **sanitize**—no complicated disassembly.

Components (scale, printer) need to be **durable** for repeated use.

Labels and adhesives should **withstand freezing and thawing**.

7. Power/Battery Life

Must offer **long battery life** between charges.

Include **low battery indicators** (LEDs, app notifications).

Quick and **safe recharging** (USB-C or other standard).

8. Connectivity

Bluetooth/Wi-Fi for app communication, but needs to be **secure** and **stable**.

Optional **offline functionality** if internet isn't available.



How You Can Present It in Your Report

Example Outline for Chapter 4:

Introduction

Purpose of standards and addressing constraints.

Applicable Standards

Health & Safety (FDA, HMBANA, ISO)

Labeling & Traceability (HMBANA, GS1)

Electronic Safety (UL, FCC)

Software (HIPAA, ISO/IEC)

Battery Safety (IEC)

Design Constraints

Portability and Power

Usability (One-handed use, UI simplicity)

Accuracy and Precision

Environmental (Temperature, moisture)

Cost considerations

Maintenance & Durability

Connectivity and Data Security

E.3: **“Help me find reference schematics**

To assist you in developing schematics for your breastmilk tracking and labeling device, here are some reference materials for the key hardware components:

1. Load Cell with HX711 Amplifier

To measure the weight of breastmilk accurately, you'll utilize a load cell in conjunction with the HX711 amplifier. [Random Nerd Tutorials+1Just Do Electronics+1](#)

- **SparkFun's HX711 Load Cell Amplifier Hookup Guide:** This comprehensive tutorial provides detailed instructions on connecting a load cell to the HX711 amplifier and interfacing it with a microcontroller. It includes wiring diagrams and explanations of the color coding for load cell wires. [SparkFun Learn](#)
- **Arduino with Load Cell and HX711 Amplifier Tutorial:** This guide walks through creating a digital scale using an Arduino, load cell, and HX711 amplifier. It covers wiring, calibration, and code examples. [Random Nerd Tutorials](#)

2. Microcontroller (ESP32) Integration

The ESP32 microcontroller will serve as the central processing unit, handling data from the load cell and controlling other peripherals. [Circuit Designer Tutorials+1GitHub+1](#)

- **ESP32 Hardware Design Guidelines:** Espressif's official document offers detailed insights into designing circuits with the ESP32, including recommended schematics for power supply, crystal oscillators, and other essential components. [Espressif Documentation](#)
- **Custom ESP32 Board Design Tutorial:** This step-by-step guide demonstrates how to design custom ESP32-based boards, which can be instrumental in understanding the integration of various components. [Instructables](#)

3. Thermal Printer Integration

For printing labels, integrating a thermal printer with the ESP32 is essential.

- **IoT Thermal Printer Using ESP32:** This video tutorial showcases the integration of a thermal printer with the ESP32, providing practical insights and common troubleshooting tips. [Circuit Designer Tutorials+2YouTube+2Reddit+2](#)
- **ESP32-Controlled Thermal Printer Project:** This project details how to control a thermal printer remotely using an ESP32, including insights into interfacing and communication protocols. [Robotics Hub](#)

4. Power Management

Efficient power management ensures the device operates reliably, especially when battery-powered.

- **ESP32 Hardware Design Guidelines:** Beyond general design, this document provides specific recommendations for power supply designs suitable for the ESP32. [Espressif Documentation](#)

By studying these resources, you can develop a comprehensive understanding of the schematics required for each component of your device. Ensure to adapt the reference designs to fit the specific requirements and constraints of your breastmilk tracking and labeling system.