

Final Design Presentation

P.U.M.P.E.D

Precision Weighing, User-Friendly Heating, Mobile App Integration, Printed Labels, Expiration Tracking, and Data Management

Group 16

Zachary Celentano

Jason Devito

Jamieson Foley

Isaiah Robinson

Sponsors

Victoria Rodriguez

Matthew Aberman

Advisors

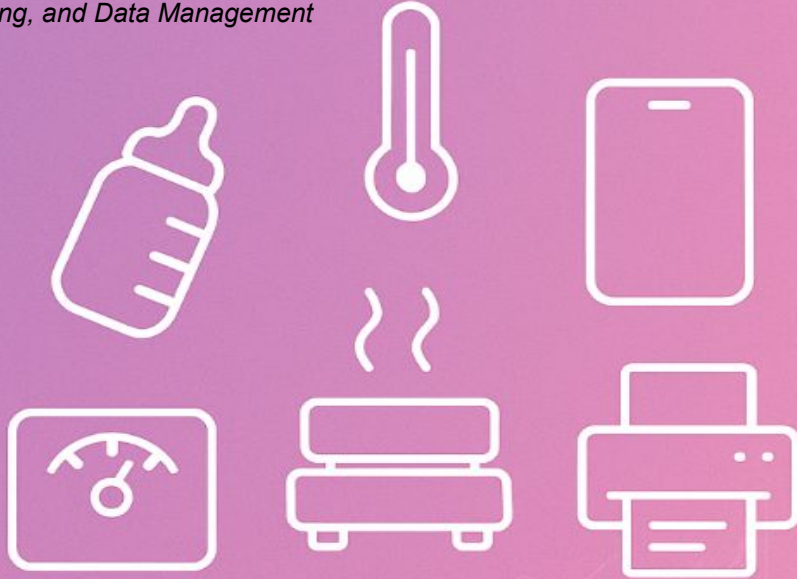
Dr. Zakhia Abichar

Dr. Wei Sun

Dr. Suboh Suboh

Mentor

Dr. Chung Yong Chan

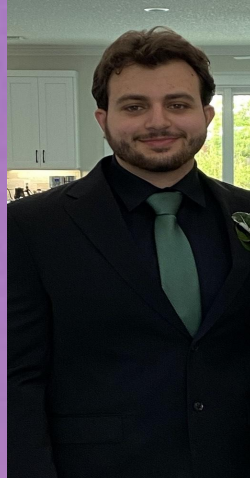


MEET THE TEAM

Zachary Celentano
Electrical Engineering



Jason Devito
Computer Engineering



Isaiah Robinson
Computer Engineering



Jamieson Foley
Electrical Engineering



Problem and Motivation



Parents currently must weigh each milk bag, write down weight, date, and time by hand, then heat it in a separate hot-water bath—steps that are slow, prone to mistakes, and unsafe if the temperature is not monitored carefully. No single product exists to combine weighing, labeling, warming, and tracking, leaving caregivers to juggle multiple tasks and worry about wasted or expired milk.

- Manual recording is time-consuming and often leads to mislabeling
- Inaccurate dates cause milk to expire unnoticed or be discarded unnecessarily
- Heating in a hot-water bath lacks precise temperature control, risking safety

The current solutions really do not **cut** it. We need to give our mothers an **easier** time!

- Existing tools only handle one step (weighing, labeling, or storage) rather than the whole process
- Our device aims to **centralize** these operations

Goals



Basic Goals:

- Build a product that allows parents to easily weigh, label, and store milk bags.
- Build an application that allows an accessible interface with the data of bagged milk.
- Build a system that ensures milk is stored and used at safe temperatures, helping prevent spoilage or degradation of nutrients.

Stretch Goals:

- Allow allergen tracking within application.
- Allow for user data to be stored within a cloud storage database.
- Provide feedback to parents within app based on usage.
- Maintaining tracked data for a period of up to a year.

Objectives



Basic Objectives:

- Implement a load cell and ADC for accurate weight measurement. Convert the measured weight to fluid ounces for parents ease of use.
- Integrate a label printer to produce labels containing all relevant information, including fluid ounces, date, time, and temperature.
- Show real-time weight, date, and time on the LCD prior to label printing.
- Measures the temperature of bagged milk using thermal coupling.
- Implement a heating element to allow parents to more freely manage milk temperature.
- Implement a custom power supply to power the device and all peripherals.
- Any surface that makes contact with the milk bags will be able to be taken apart for sanitation purposes.
- Develop an iOS app to interface with a database of information regarding stored milk.
- Provide features for viewing, editing, and managing milk inventory.
- Send alerts for impending expiration dates and low general storage.

Stretch Objectives:

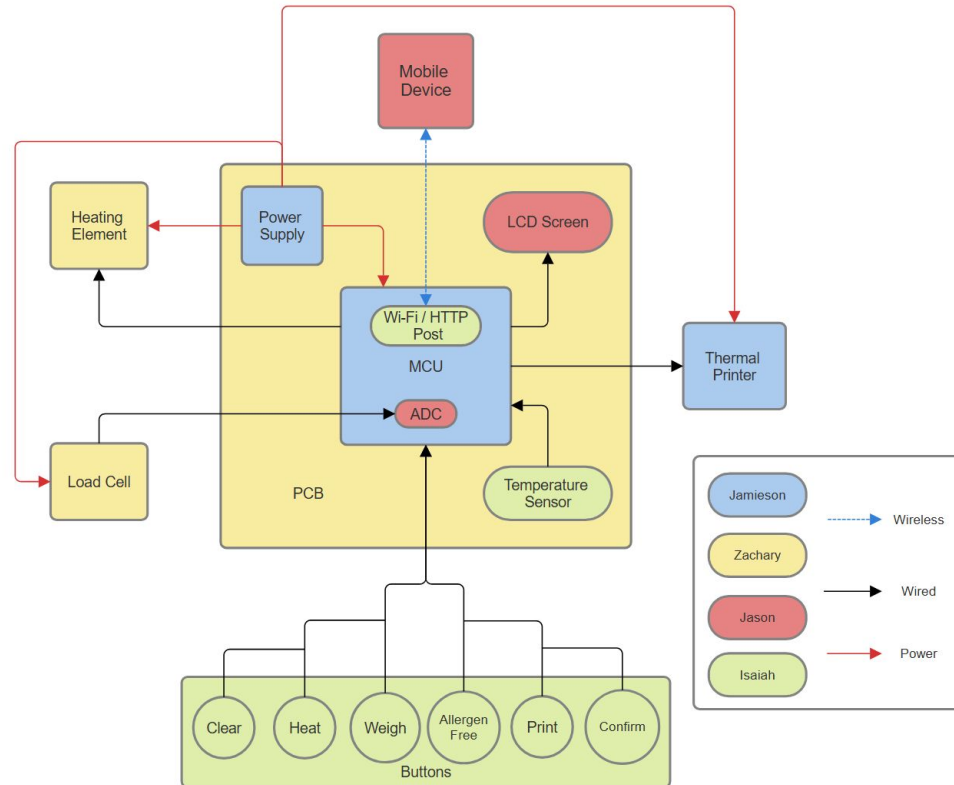
- Build a custom thermal printer and its subsequent PCB to minimize form factor.
- Design the power supply to be able to work with batteries so the product can be portable.

Engineering Specifications

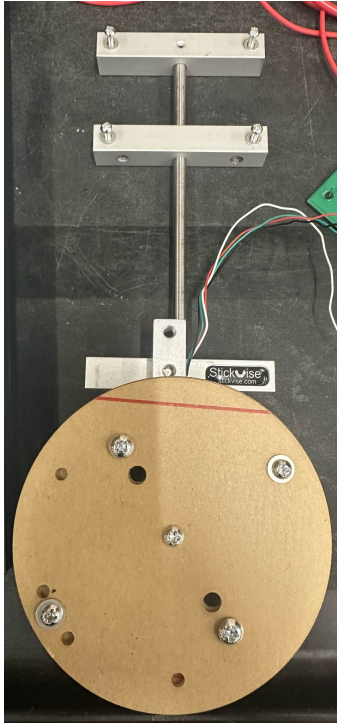


Parameter	Value	Description
Weight Measurement Accuracy	± 10 Grams	Accurate weight measurements.
Temperature Measurement Accuracy	± 1 °C	Accurate temperature measurements.
Label Size	2" x 1"	Large enough to display all data.
LCD Display Time	< 1 Second	Fast response time.
WiFi Connectivity Response Time	< 5 Seconds	Efficient interface connectivity.
Print Time	<7 seconds	Fast label creation
Local Data Storage Capacity	> 20 entries	Enough data storage capabilities for a large inventory
Heating Temperature	90-100 °F / 33-38 °C	Must heat water to the proper temperature to warm up milk
Heating Time	<30 minutes	Must heat in an appropriate time limit
Cost Estimate	< \$800	Needs to be cost effective to manufacture.

Hardware Block Diagram



Physical Product



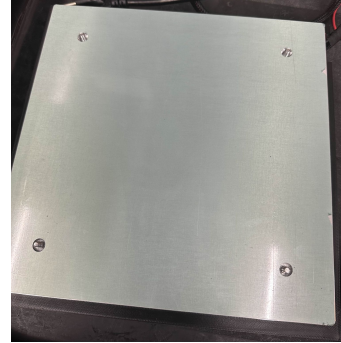
Load Cell



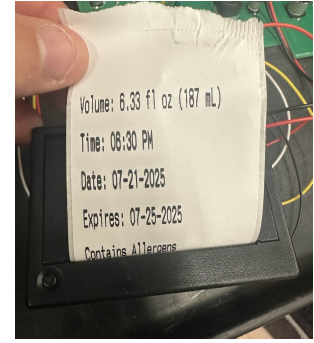
Enclosure w/ Transformer and Rectifier circuit



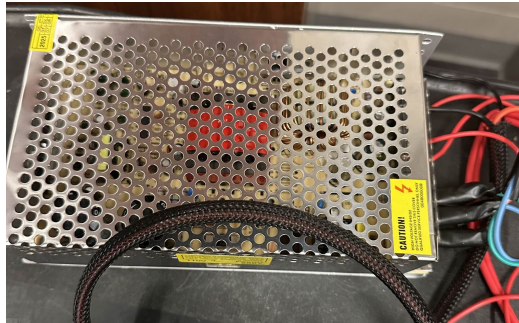
Thermocouple



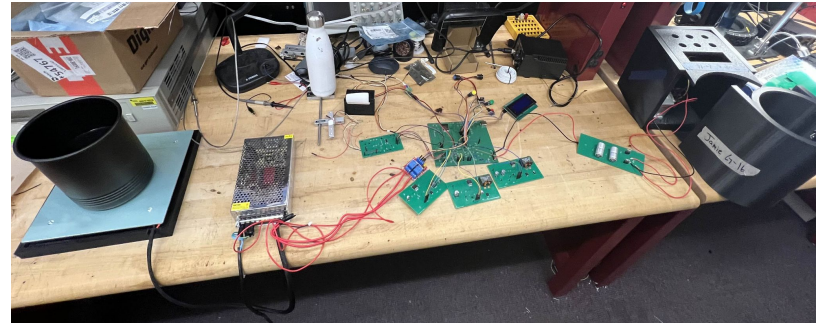
Heating Pad



Thermal Printer



Power Supply Adapter Transformer Switch



Entire Product (Laid Out)

Hardware Comparison: Weight Measurement



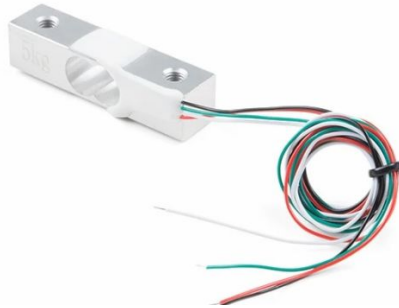
- High accuracy with little re-calibration over time
- Varying prices for load cells
- Relatively simple to implement.

Method	Cost	Implementation	Accuracy
Load Cell	Moderate (Anywhere between \$ & \$\$\$)	Moderate (Requires an amplifier IC)	High (Does not lose accuracy over time)
Piezoelectric Force Sensors	High (Between \$\$ & \$\$\$)	Difficult (Requires Charge amplifier)	Moderate (Loses accuracy with time, best for alternating forces)
Capacitive Force Sensors	Low (Between \$ & \$\$)	Moderate (Requires Capacitive Sensing IC)	Moderate (Sensitive detection, best for low force detection)

Hardware Selection: Load Cell



- Includes an amplifier IC with the purchase.
- Has the lowest cost.
- Manufacturer offers the schematic design of the amplifier IC necessary for operation.



Product	Cost	Implementation
Phidgets CZL635 (5kg)	\$7 per unit + \$30 - \$95 for proprietary IC	Easy (Can implement amplifier IC into PCB design)
SparkFun SEN-14729 (5kg)	\$13.12 per unit	Moderate (Requires amplifier design and implementation)
TAL220B (5kg)	\$11.95 per unit	Moderate (Requires amplifier design and implementation, which is provided by the seller)

Hardware Comparison: Label Printer



- Requires the least amount of maintenance.
- Has the lowest cost.
- Greatest disadvantage is finding one that will easily be implemented into our device.

Label Printer	Cost	Implementation	Maintenance
Thermal printer	Low \$50-\$100	Easy	Low (fewest moving parts, no toner or ink)
Laser Printer	High \$100-\$500	Moderate	Moderate (toner needs to be replaced often. Regular cleaning needs to take place)
Inkjet Printer	Moderate \$50-\$300	Easy	Moderate (regular cleaning of the printhead to avoid clogging)

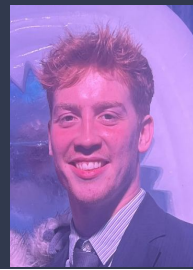
Hardware Selection: Thermal Printer



- Cost is a primary concern
- Most thermal printers work the same
- Chose the ASPIRON for its lowered cost and moderate integration
- Integration is through serial port

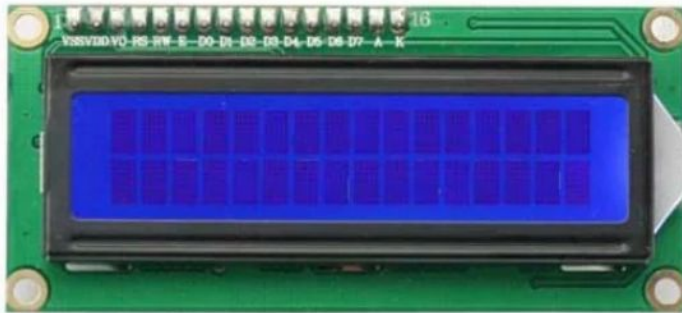


Thermal Printer	Cost	Integration	Maintenance
MUNBYN IMP001 58mm Thermal Printer (Serial/UART & USB)	Moderate \$40-\$80	Moderate as it uses UART or USB connection	Moderate maintenance as needed depending on use of paper, also dependent on jams.
PRIMUZ Micro Thermal Printer	Moderate \$45-\$65	Moderate, can be connected through USB or through serial port	Ventilation may be a concern, as overheating could possibly occur.
ASPIRON 58mm Embedded Thermal Printer Module (UART & TTL Interface)	Moderate \$45-\$70	Moderate; connected through USB or through serial port	Similar ventilation problems are possible, also may need additional protection from dust buildup.



Hardware Selection: Display

- Has lowest cost and is the easiest to integrate into our system.
- Requires low maintenance as it is durable with simple graphics.
- Has the option of being upscaled to a 20x4 character LCD with little to no changes to be integrated into our system.



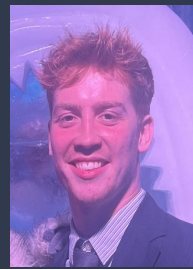
LCD Display	Cost	Integration	Maintenance
16x2 Character LCD (HD44780 - I2C or Parallel Interface)	Low \$3-\$10	Easy integration with simple character display and uses I2C	Low as it is durable with no complex graphics. Has minimal issues
128x64 Monochrome LCD (ST7920 - SPI/I2C/Parallel Interface)	Moderate \$6-\$15	Moderate integration and requires SPI/I2C/Parallel interface. Also needs a graphics library to support simple graphics and texts	Low as it is reliable, but backlight or contrast may degrade overtime
2.8 inch TFT LCD (ILI9341 - SPI Interface, Full Color)	High \$10-\$30	Complex integration, requiring graphics library, more memory and processing power needed	Moderate as it is more fragile. Color display may degrade and the touchscreen may wear out

Hardware Comparison: Heating



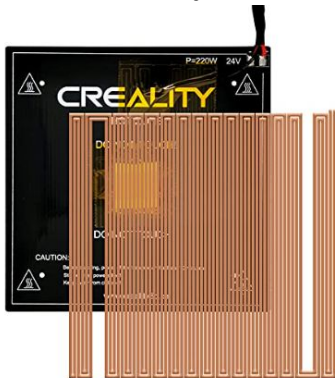
- Conduction has the highest proficiency in heating
- Operational cost is higher, but efficient in heating an item directly.
- If conduction is efficient, convection will be present in the water
- Efficiency will increase if both are working

Method	Efficiency	Implementation
Conduction	High (Element makes direct contact with subject, and direct heat transfer is better for small area heating)	Easy as the heating element is directly in contact with the water which would transfer the heat efficiently
Convection	Moderate (heats large spaces well, requires consistent fluid motion)	Moderate (implementation of forced convection would require moving parts to move fluid)
Induction	High (requires significant space for coils generating magnetic current, energy efficient)	Difficult (requires complex power management due to AC requirement for operation)



Hardware Selection: 3D Printer Heating Bed

- Selecting the 3D Printer Hot Bed
- The heating option will not burden the customer
- Simple to integrate and maintain
- Powered through switching PSU and relay

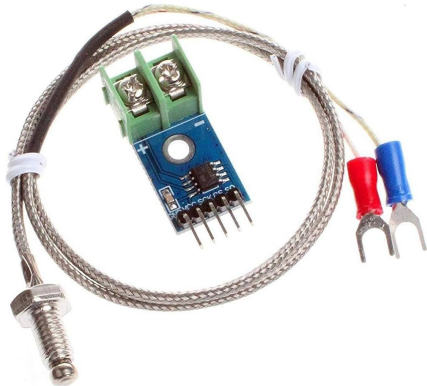


Heating Pad	Cost	Integration	Maintenance
Electric Heating Pad 14cm x 5cm by Adafruit	Low \$6	Easy to integrate as it is thin and flexible and can be attached to a surface by an adhesive. Requires MOSFET or Relay for control	Low maintenance as there are no moving parts. The adhesive will weaken over time
Official Ender 3 V2 Heated Bed 3D Printer Replacement Hot Bed	Medium \$22	Easy to integrate as it is Requires MOSFET or Relay for control. Simple heating pad that lays on surface	Low maintenance as there are no moving parts. Very durable choice.
Cartridge Heater by MakerTechStore	Low \$4	Moderate to integrate as it requires a hole or slot to be inserted into a heating block or reservoir of water. Requires MOSFET or Relay for control	Medium maintenance as it can accumulate limescale. Requires regular cleaning.

Hardware Selection: Temperature Sensor



- Moderately priced but makes up for it in efficiency
- Simple and easy integration
- Low maintenance due to minimal upkeep
- Cleaning of this sensor will need to occur so that the sensor does not degrade from outside elements



Temperature Detector	Cost	Integration	Maintenance
Type K Thermocouple with MAC6675 Amplifier	Moderate \$4-\$13	Easy to integrate. Uses SPI making it easy to connect to most microcontrollers	Low, durable with minimal upkeep. Cleaning will need to occur to prevent degradation of the sensor.
PT100(RTD-Resistance Temperature Detector)	High \$10-\$100+ Varies significantly with accuracy and material	Moderate to Complex. Requires an excitation current, specialized amplifier and signal processing	Moderate to High as it is more fragile and needs careful wiring. Needs to be calibrated overtime
NTC Thermistor (Negative Temperature Coefficient Sensor - 10K Ω or 100K Ω)	Low \$0.50-\$5	Easy to Moderate. Requires voltage divider, but is more sensitive to noise	Moderate as it is prone to aging and drift. This also needs to be recalibrated over time

Hardware Selection: Microcontroller



- The ESP32 is the ideal choice as it provides the most features at an affordable cost
- It enables both bluetooth support as well as Wifi support, which is ideal for our MCU of choice



Name	WiFi	Bluetooth	RAM	Clock Speed	Cost
ESP32	Yes	Yes	520 KB	240 MHz	Low (\$ per unit)
STM32	No	No	512 KB - 1MB	84-216 MHz	High (\$\$ per unit)
nRF52840	No	Yes	1 MB flash, 256 KB RAM	64 MHz	Medium (\$-\$ per unit)

We plan to use buttons for the user interaction as they are able to be used in multiple different ways depending on each function of our system

Power Distribution



Device	Voltage	Current (Typical)	Power (Peak in W)
ESP32-WROOM	3.3 V	240 mA	0.792 W
LCD Display	5 V	25 mA	0.125 W
Load Cell + Amplifier	3.3 V	1.5 mA	0.0075 W
Thermal Printer	12 V	2.5 - 3 A	30 - 36 W
Thermocouple	5 V	60 μ A	0.0002 W
5V Dual Relay	5 V	140 mA	0.7 W
3D Printer Heating Bed	24 V	9-10 A	216 - 240 W

Hardware Comparison: Power Supply



- The primary factor in the power design for our system is cost effectiveness.
- Utilizing grid power minimizes the necessity for additional infrastructure.
- Chargers for batteries or solar panel installations are not needed.
- Grid power can readily provide the larger current draw needed to heat the water

Power Supply	Cost	Integration	Maintenance
Grid Power	\$5-\$50, depending on voltage and current requirements	Requires a power jack and voltage regulation circuit	Low, as it relies on existing power infrastructure
Battery Storage	\$20-200, depending on capacity, chemistry, lifespan and energy needs	Requires charging circuit, battery management system and protective enclosures	Medium, because of charging cycles and the battery degrading over time
Solar with Battery Storage	\$50-\$500, depending on solar panels, charge controllers, a battery storage system, and power output	Requires solar panels, charge controllers, and power management circuitry	High, because of environmental exposure the solar panels and batteries degrade overtime

Hardware Selection: Power Supply



- Using Grid Power infrastructure, we need to turn the 115 Volt outlet output to a usable power.
- To do this, we are choosing a signal transformer.
- The signal transformer drops the initial voltage which will input to a rectifier circuit that outputs approximately 15 volts
- This option is more expensive, but will afford us modularity in our design if power needs change.



Product	Cost	Implementation	Maintenance
Bel Signal Transformer with Rectifier Circuit	Moderate (\$30-\$50)	Hard (We will have to self build everything to our design specification)	High (with a lot of individual parts, there is a possibility of having to fix individual parts if problems arise)
LRS-100-24 (SMPS Adapter)	Low (\$17)	Easy to Moderate (Comes pre-built)	Moderate (cleaning may need to occur to prevent dust build up)
Alitove Power Supply(AC/DC Adapter)	Low (\$23)	Easy to Moderate (Comes pre-built)	Moderate (cleaning may need to occur to prevent dust build up)

Secondary Power Supply



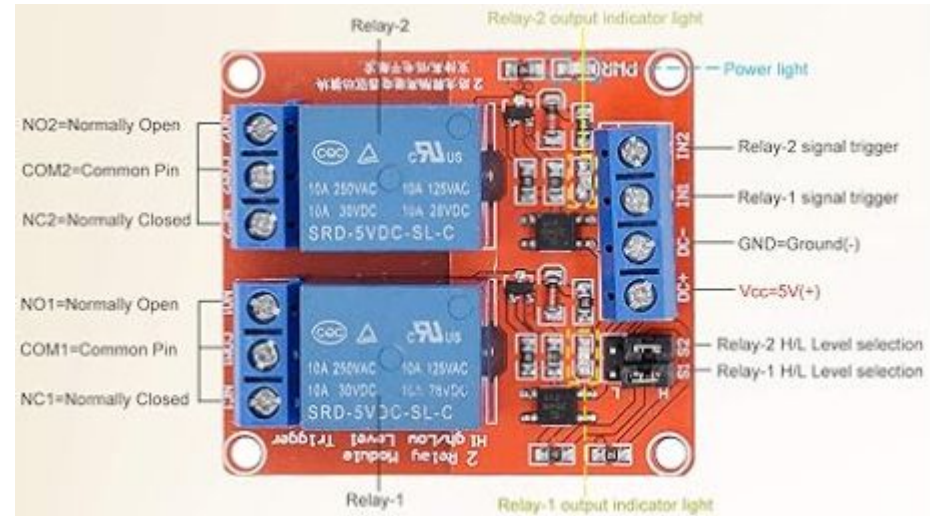
- Power Supply Adapter Transformer Switch
- AC 110V / 220V
- Output of 24V, 10A, 240W
- Chose out of necessity to power heating pad
- Is in connection to the MCU through a 5V dual relay



5v Dual Relay Module



- DC 5V, 2 Channels
- Includes a High/Low trigger
- Load range of DC 30V/10A
- Each Relay will handle 24V/5A
- Relay is controlled through 5V logic signals from microcontroller



Hardware Comparison: Linear v. Switching Regulators

Type	Design Complexity	Integration	Cost
Linear	Low	Easy	Low
Switching	Moderate	Moderate	Moderate

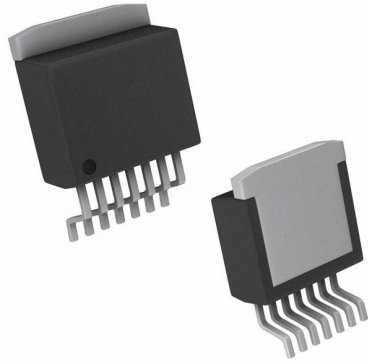


- With regards to regulators, we need regulators that can handle significant current from our supply and potentially output larger amperage.
- Switching regulators, while generally more expensive and complex in design, are the more viable choice because they can handle more current.

Hardware Selection: Switching Regulators

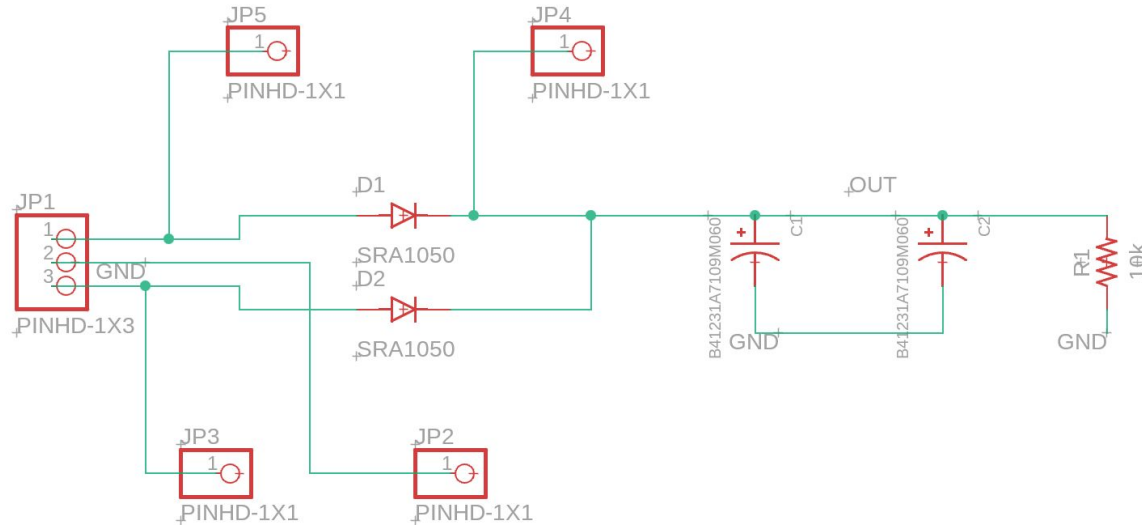


- At an affordable price we selected the LM2679 as it will perform with efficiency
- It also has a low integration level at an affordable cost



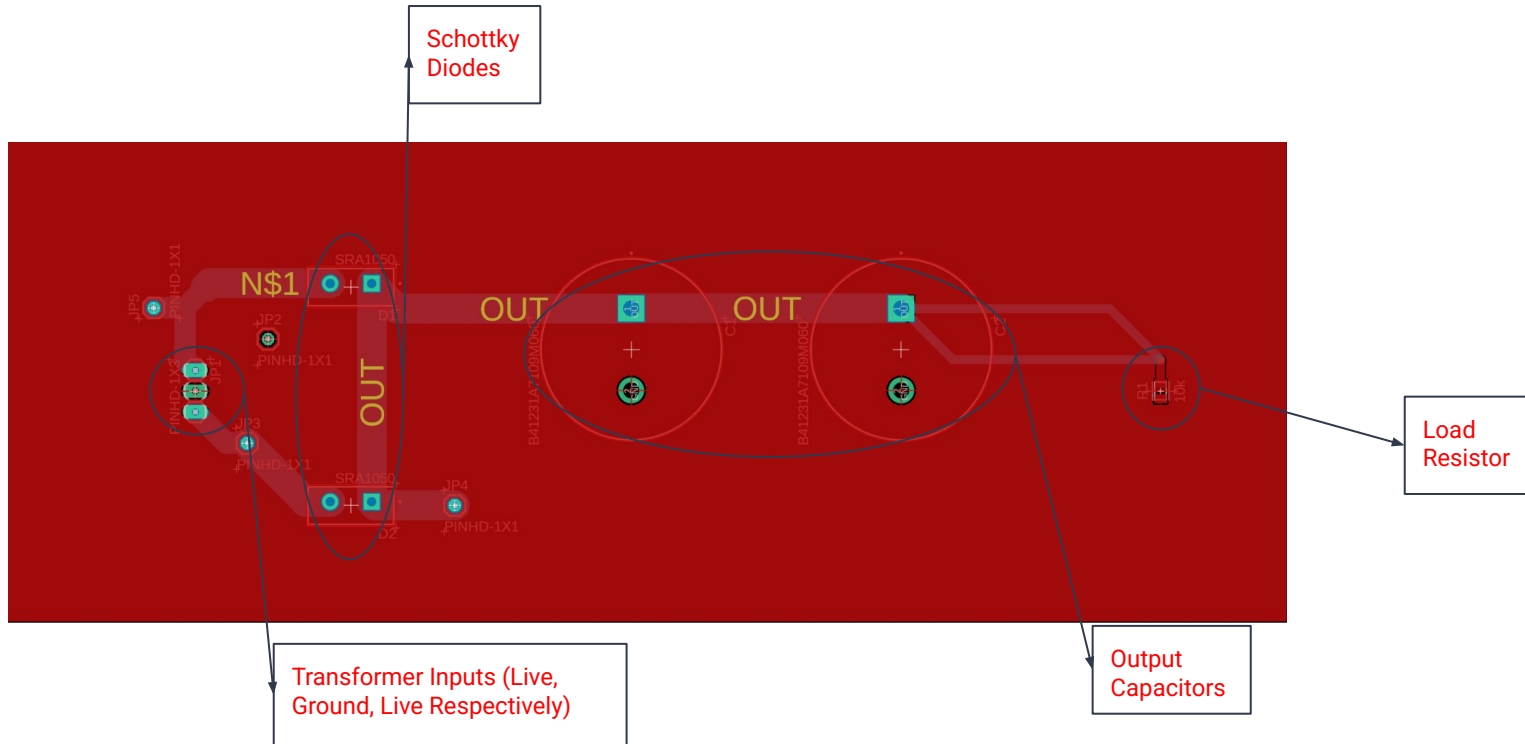
Regulator	Integration Difficulty	Cost
LM2679	Low	Medium
MP2322GQH-Z	Moderate	Low
LTC3355EUF#TRPBF	High	Medium

Hardware Diagram: Rectifier



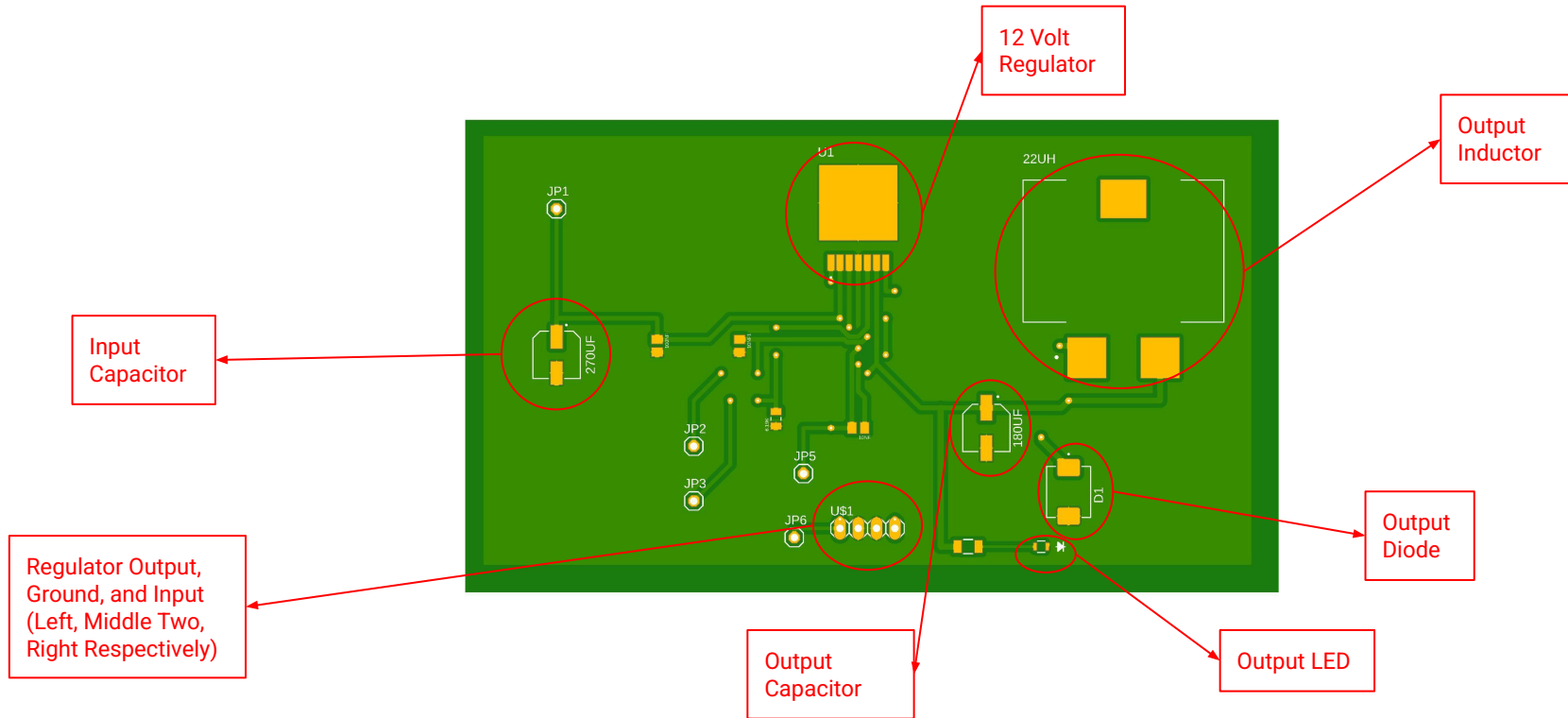
- The noteworthy components in this schematic are the schottky diodes and capacitors.
- We are utilizing a full wave, center tapped rectification circuit.
- The two outputs of the schottky diodes will give us a rectified output and the capacitors will be significantly large as to “smooth” the output response.

PCB Rectifier

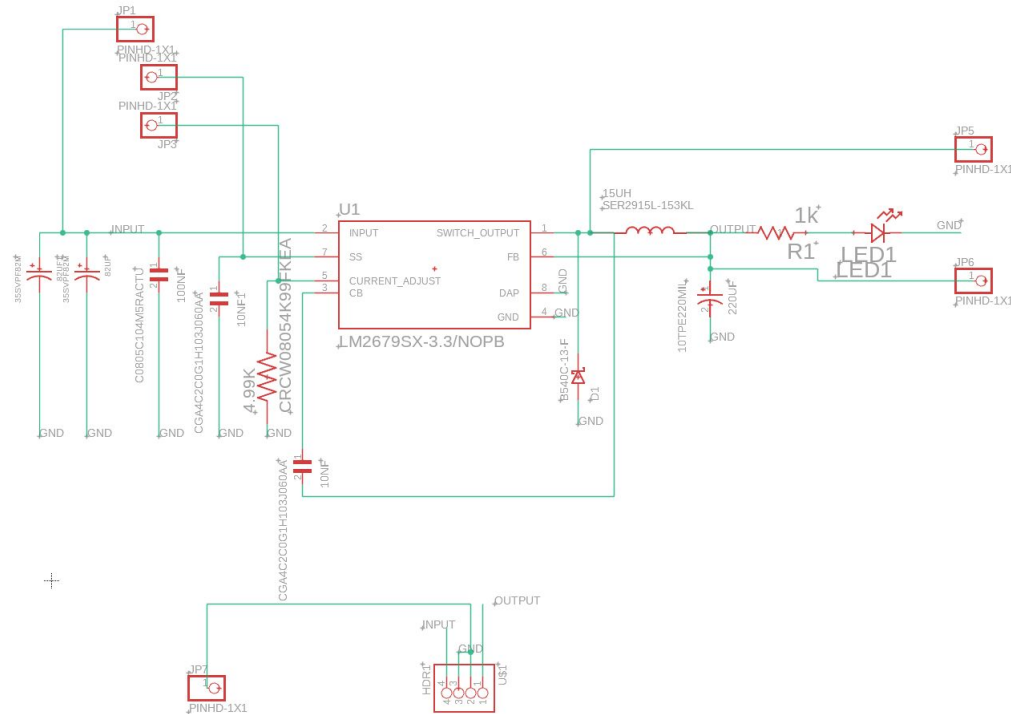




PCB Regulators: 12 Volt

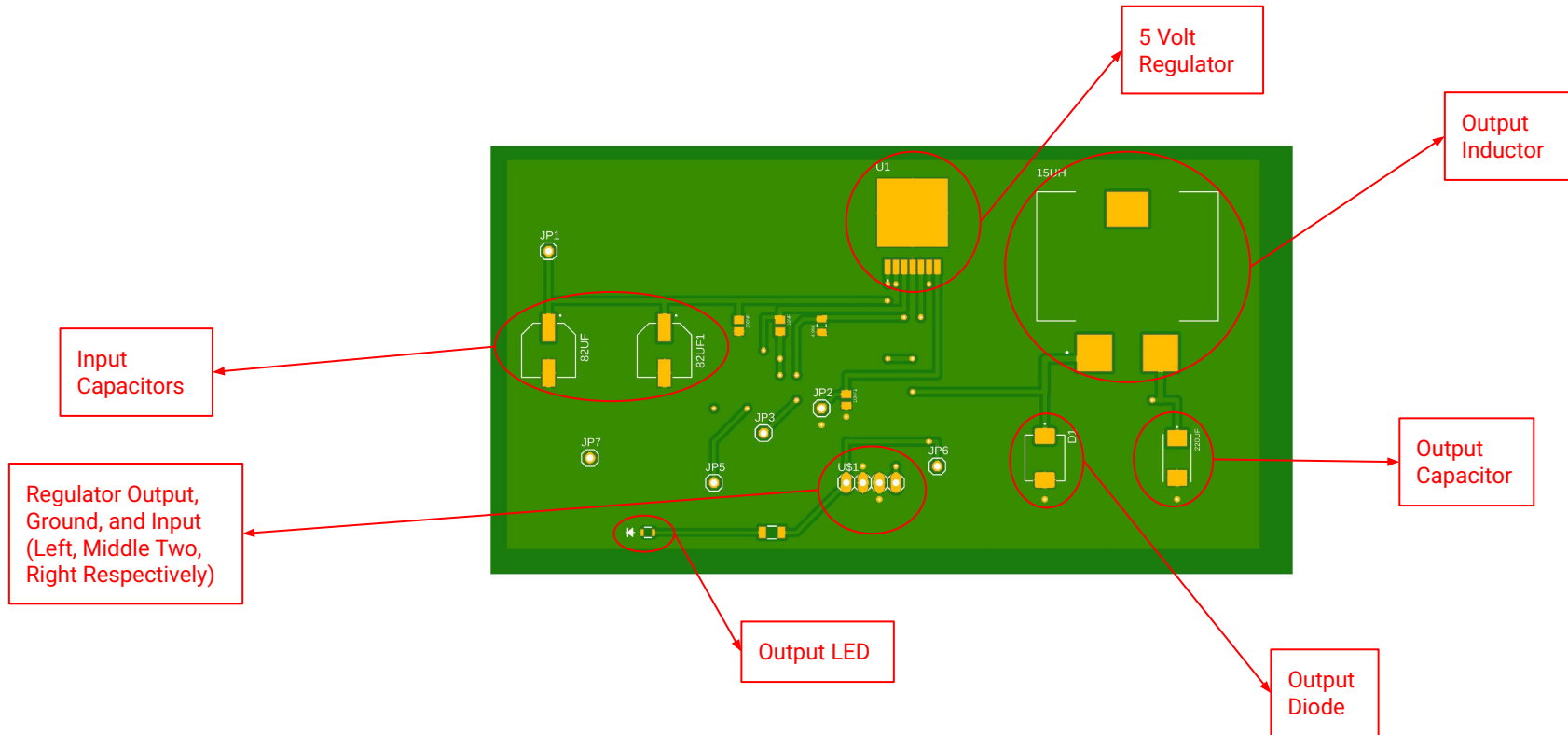


Hardware Diagram: 5 Volt Regulator

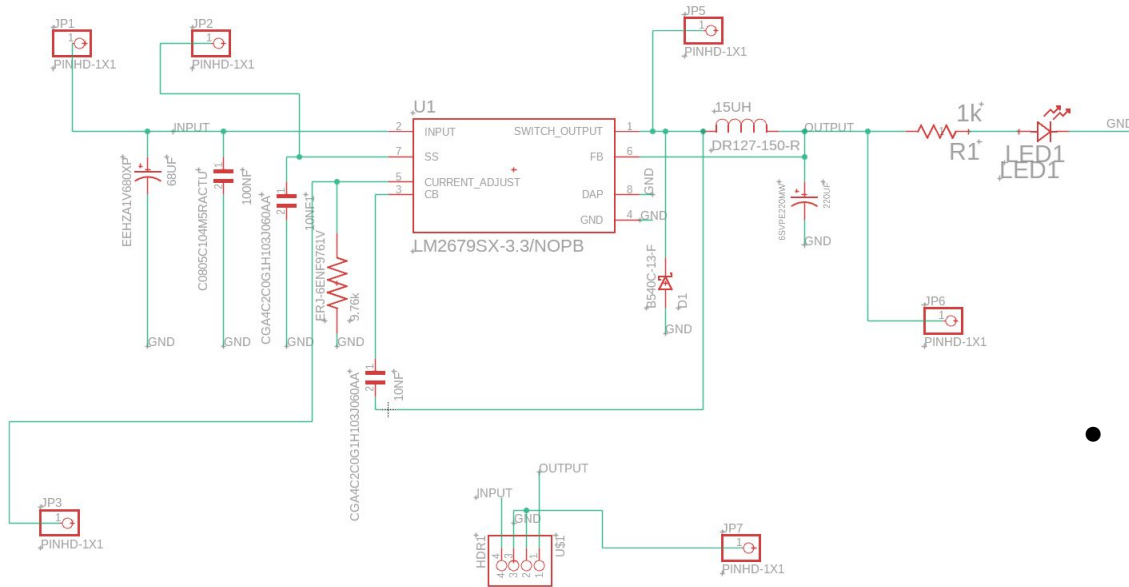


- Power supply for LCD, Buttons, Relay, and Temperature Sensor

PCB Regulators – 5 Volt

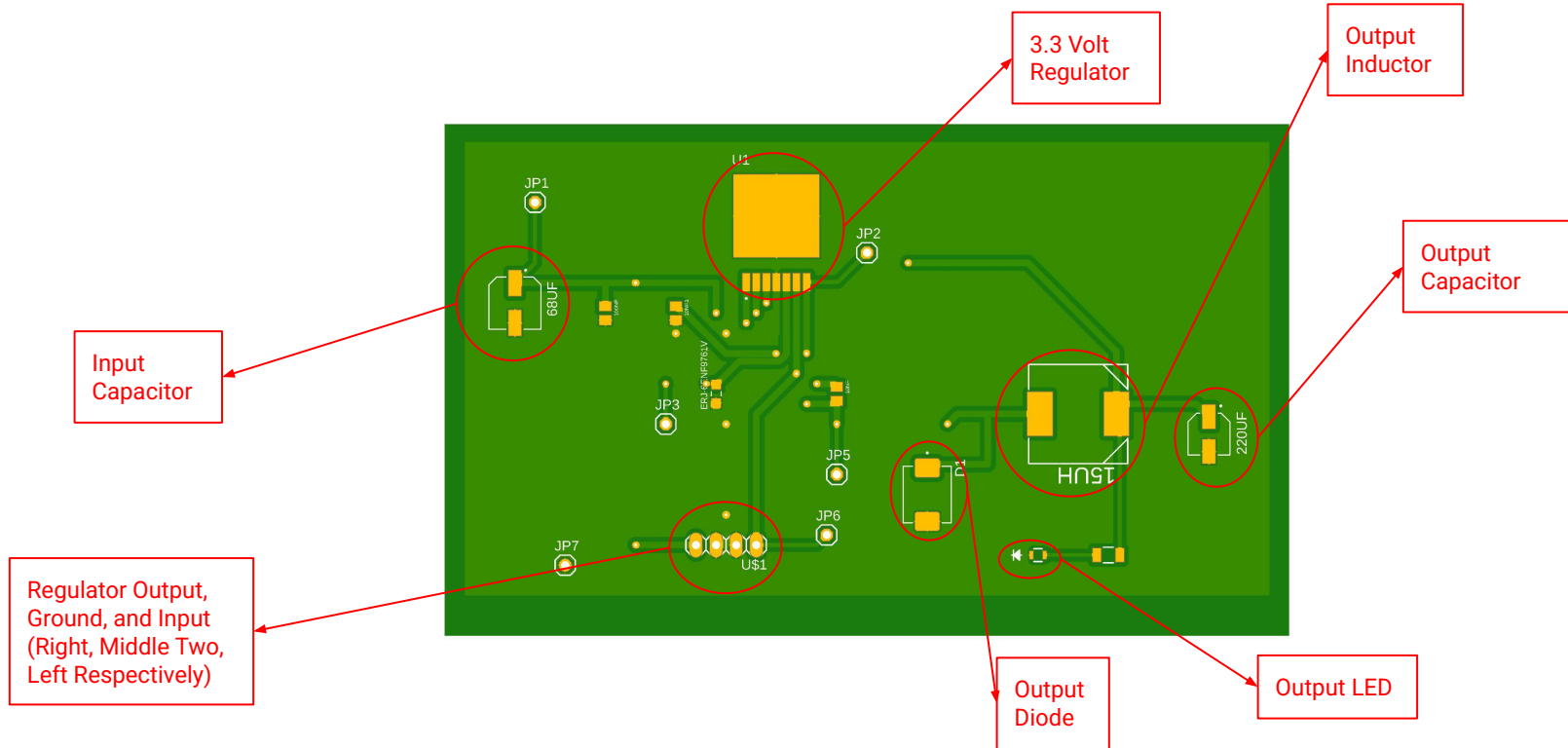


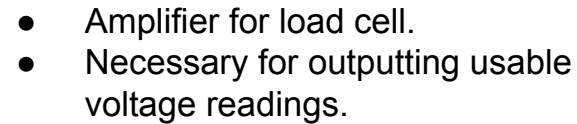
Hardware Diagram: 3.3 Volt Regulator



- Provides power to MCU, and load cell amplifier

PCB Regulators – 3.3 Volt





PCB Amplifier

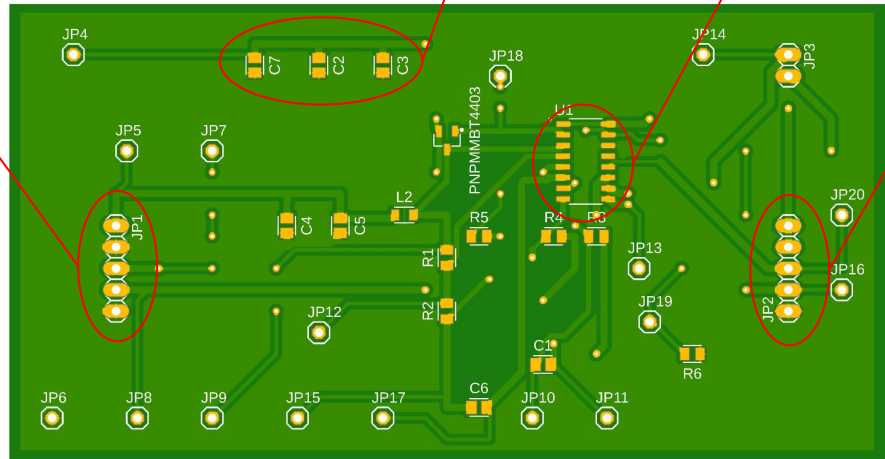


Load Cell
Connections

Decoupling
Capacitors

HX711

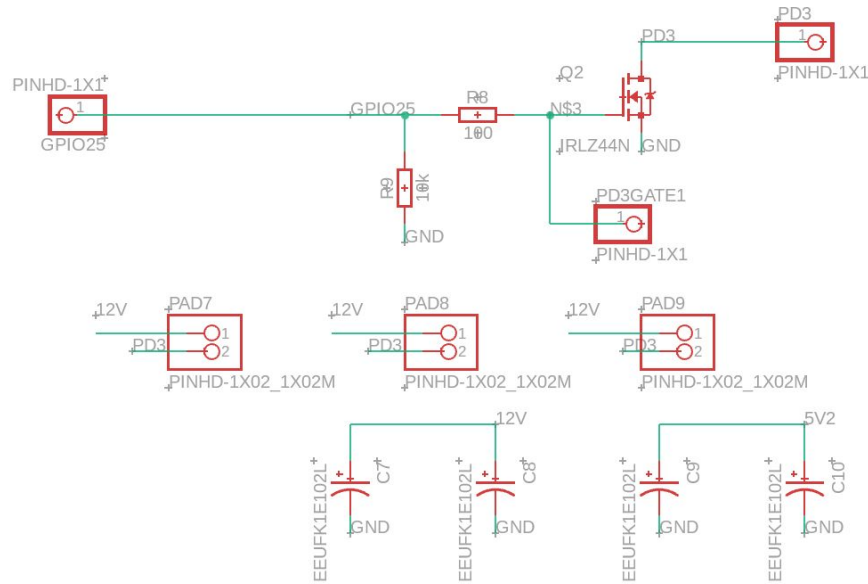
ESP Connections





- 35

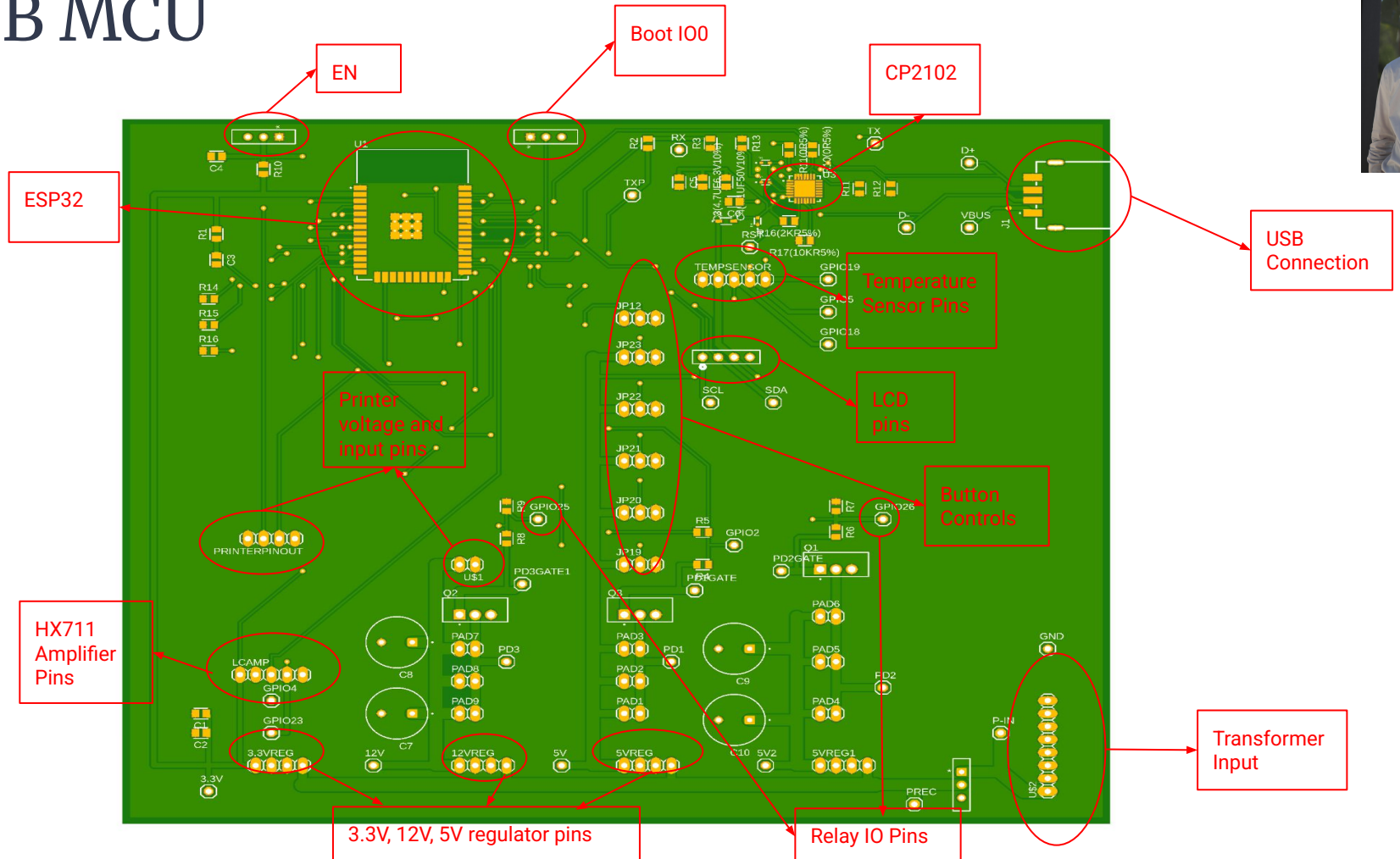
Hardware Diagram: MCU Cont.



- This section of the MCU is a separate MOSFET driver circuit, as well as decoupling capacitors and connection points

-

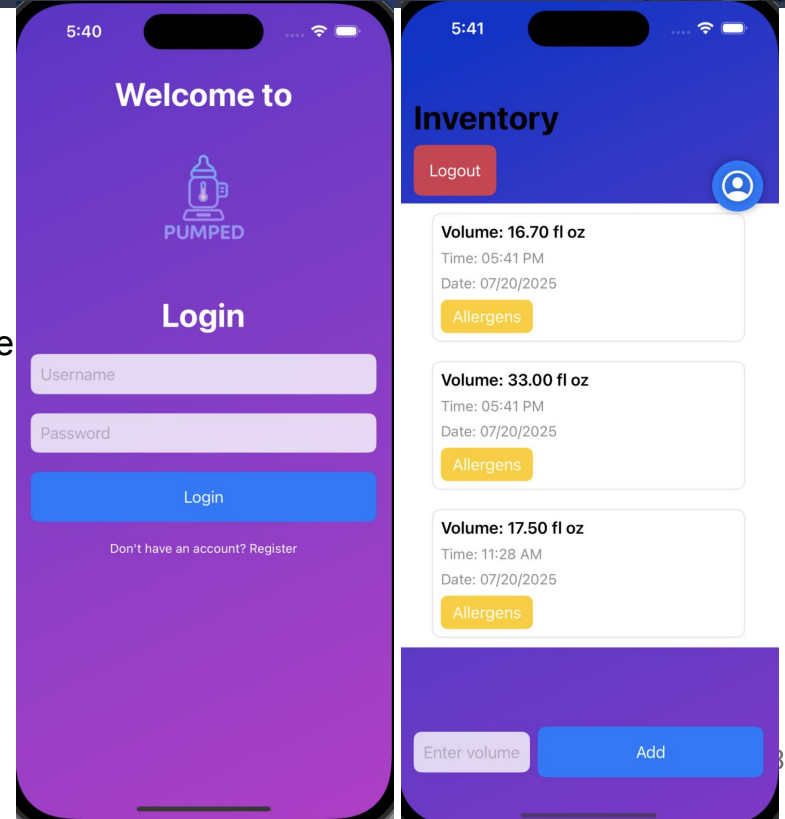
PCB MCU



Mobile Application



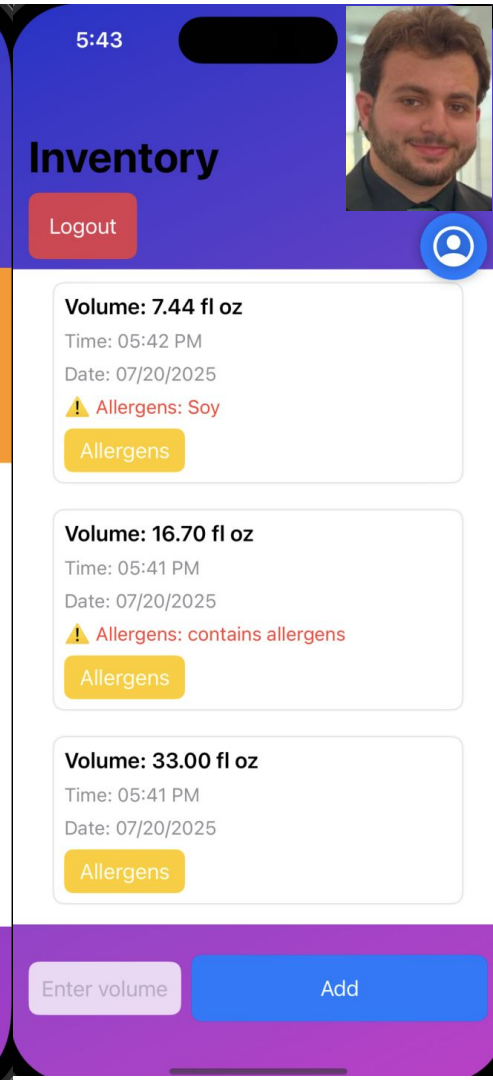
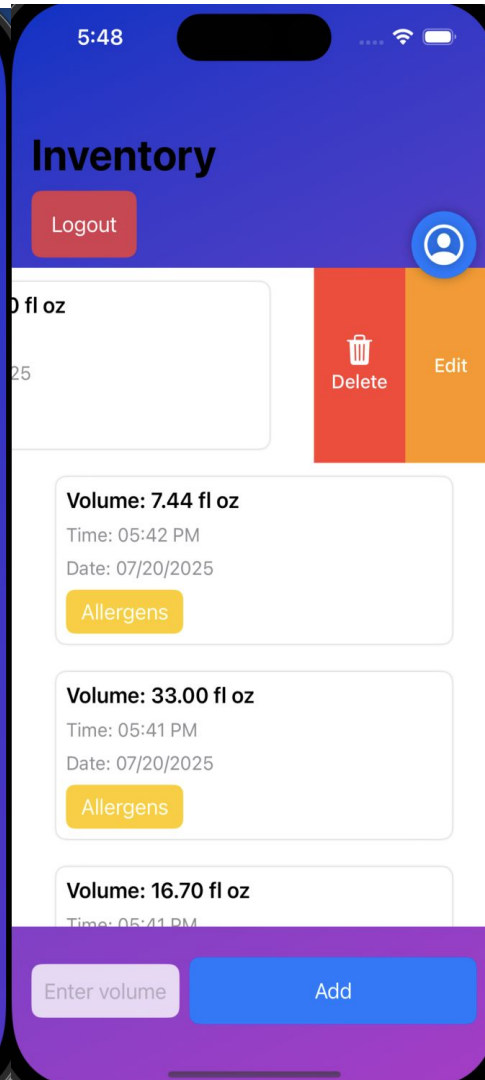
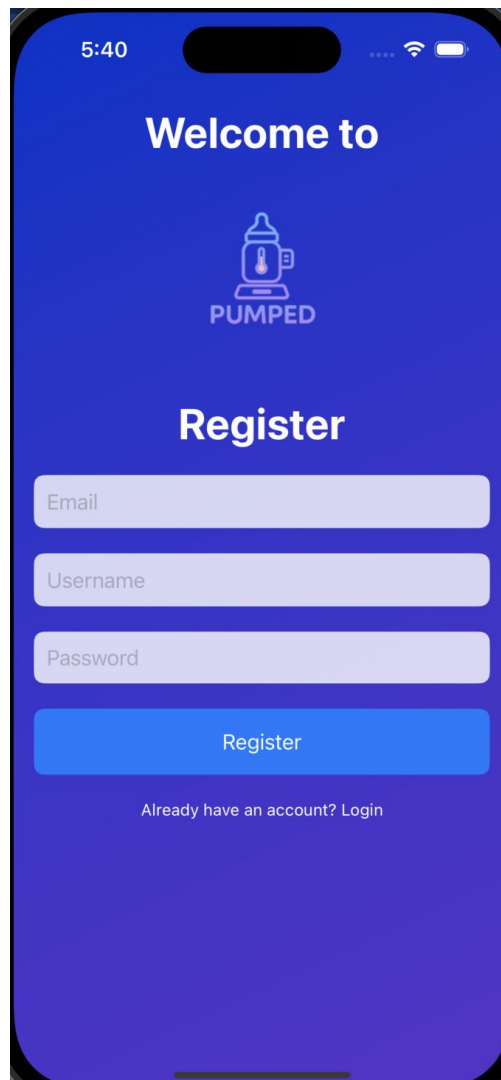
- **User Authentication**
 - Login and Register
- **Supply Dashboard**
 - Fetch measurements from backend
 - Graphical Interface with current supply
 - Allergen button
- **Manual Management**
 - Can add, edit, and delete measurements for convenience
- **AI Powered Agent**
 - ChatGPT LLM tuned as “PUMPED” assistant
- **Notifications**
 - Push alerts when inventory reaches 3 measurements
- **User Interface**
 - The interface is intended to be simple, yet reactive and colorful
 - Custom logos implemented and generated by ChatGPT



The images on the right are the finalized design

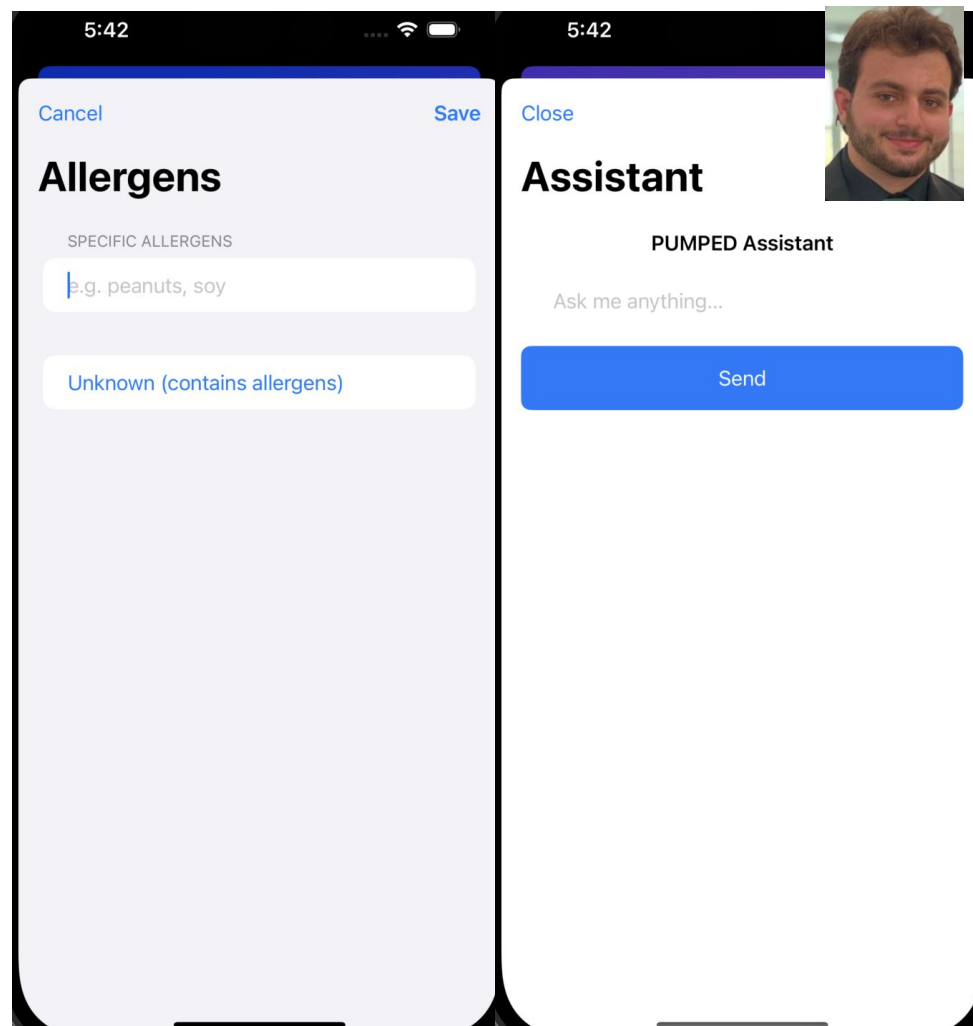
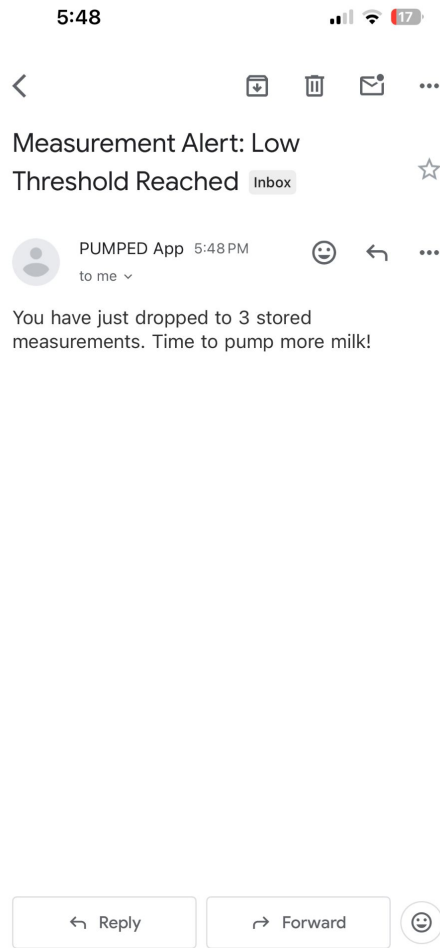
App UI

- Register Page
- Edit and Delete swipe features
- Inventory with Allergens
- AI assistant button in the top right corner



App UI

- Custom app icon
- Low inventory notifications via email
- Allergen pop up page
- AI assistant pop up page



Software Technology Comparison



We evaluated multiple options across three software layers to find the best fit for performance, maintainability, and long-term support

- **Mobile Application**

- Swift - Native iOS performance & direct access to Apple SDKs; limited to the Apple ecosystem
- Flutter/Dart - One codebase for both iPhone & Android; produces larger apps
- React Native - Leverages web skills for mobile; can feel slower with heavy updates

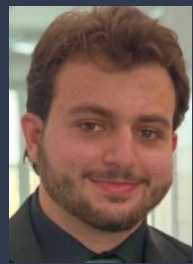
- **Backend**

- Node.js(express) - JavaScript on server for quick data handling; not ideal for heavy calculations
- Django(Python) - “All-in-one” framework with built-in admin & security; can be bulky for small changes
- Ruby on Rails - Rapid app setup with conventions; uses more server resources

- **Database**

- PostgreSQL - Strong data safety and flexible fields; more work to scale across many servers
- MySQL - Very common and reliable for straightforward data; limited flexibility for changing schemas
- MongoDB - Very common and reliable for straightforward data; limited flexibility for changing schemas

Software Technology Selection



Based on our evaluations, we selected the technologies that best balance performance, ease of development, and long-term support

- **Mobile Application**

- Swift(XCode)
 - Fast, smooth experience on iPhone
 - Sponsors wanted an iPhone only app
 - Easier to get on appstore

- **Backend**

- Node.js(Express)
 - Experience using JavaScript in past projects
 - Excellent at handling real time updates
 - Vast ecosystem of readymade packages and libraries

- **Database**

- PostgreSQL
 - Secure, reliable storage with strict data safety
 - Great for rigid structures data like our project measurements
 - Strong community and industry adoption

Backend API Routes



No code displayed—just a description of each endpoint's routes, inputs, and actions

- **User authentication**
 - *POST /auth/register*
 - Inputs: email, username, password
 - Action: Validates fields, checks for existing user, hashes password, creates new user
 - *POST /auth/login*
 - Inputs: username, password
 - Action: Validates credentials, compares password hash, sends JWT
- **Measurement management**
 - *POST /measurements*
 - Inputs: volume (in ounces)
 - Action: Captures current date/time, saves record to database
 - *GET /measurements*
 - Inputs: (none)
 - Action: Fetches all measurements ordered by newest first
 - *PUT(edit) /measurements/:id*
 - Inputs: URL param: id
 - Action: Updates an existing measurement
 - *DELETE /measurements/:id*
 - Inputs: URL param: id
 - Action: Removes the measurement record from the database
- **LLM Integration**
 - *POST /generate*
 - Inputs: userQuestion (plain-English requirement)
 - Action: Sends prompt to ChatGPT-powered service, retrieves raw gpt answer

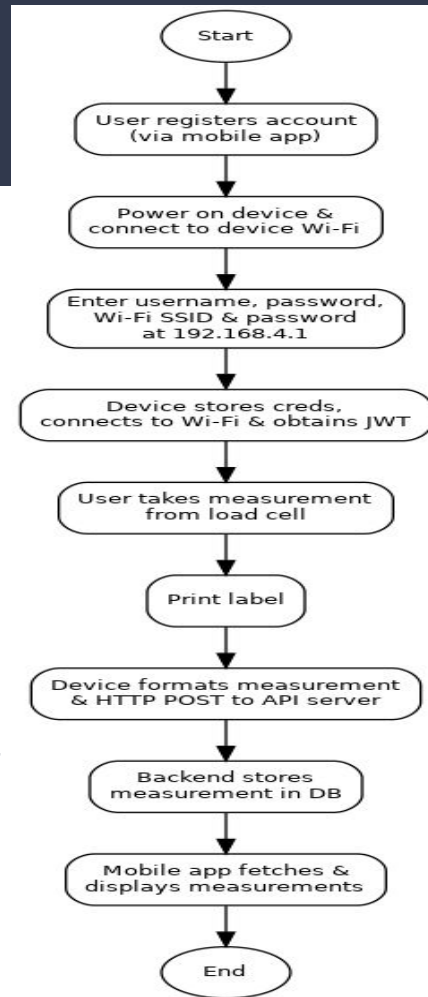
Data Flow & Handling

In order for our firmware to deliver data that is displayed on the mobile application we used some of the ESP32's built in features

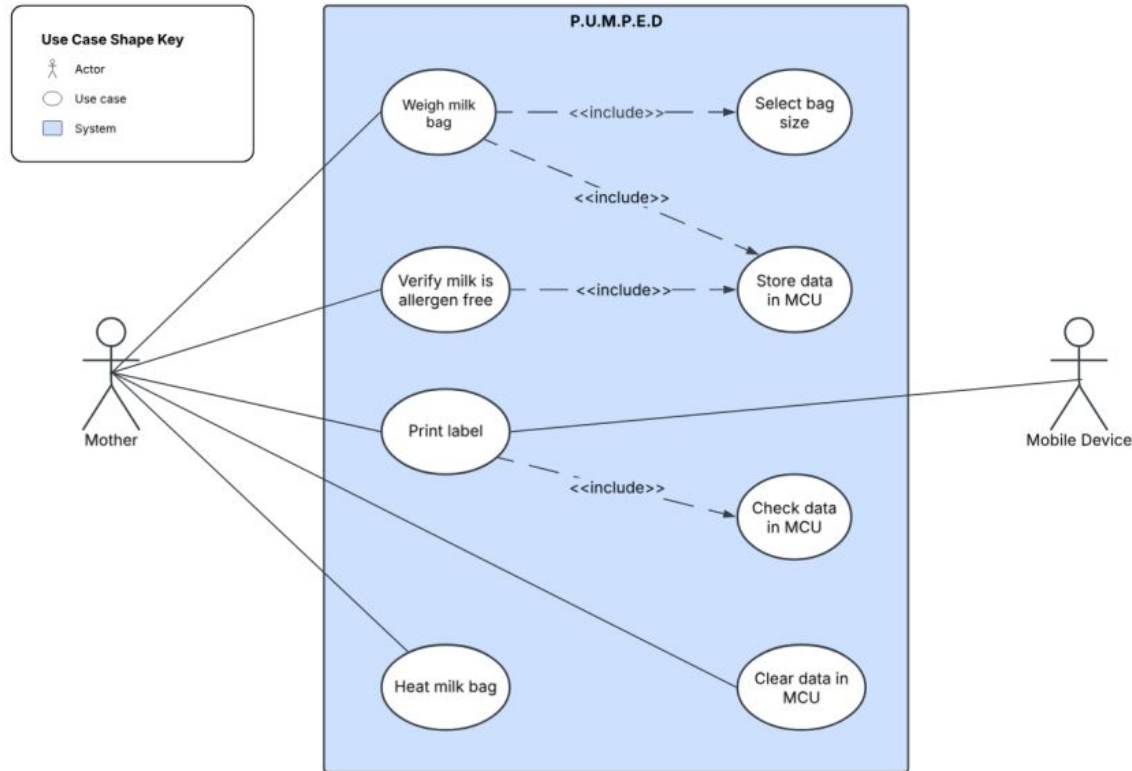
- Wifi/AP server
 - ESP spins up html server, user inputs credentials,
 - Connects to local network using `WiFi.begin(userssid, user password)`
- HTTP Communication
 - HTTPClient for simple POST requests
 - JSON payload with volume, date, and time

User and Device Data Flow:

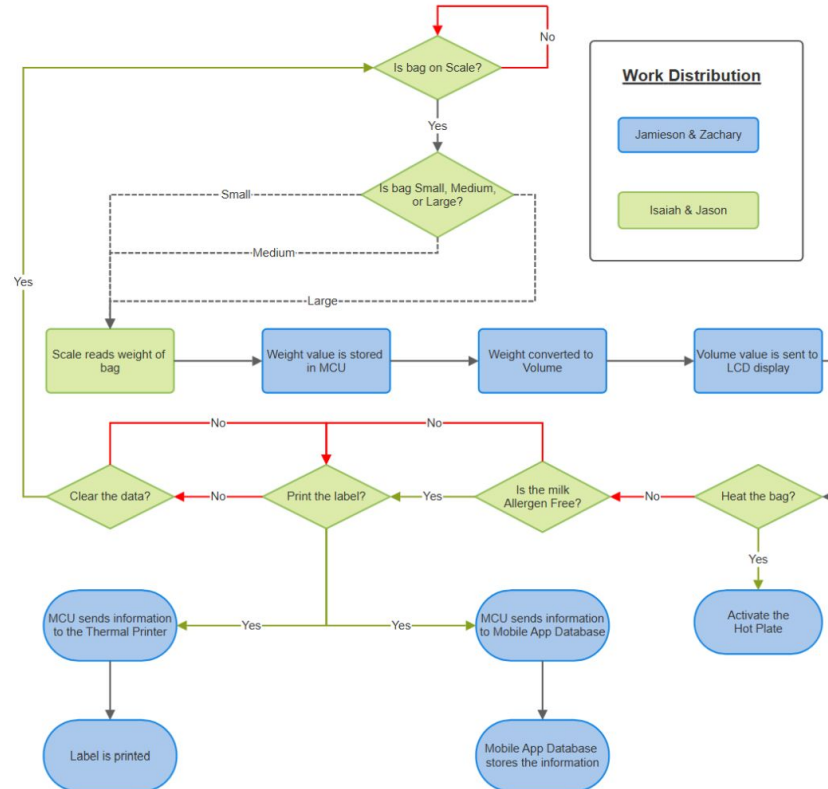
1. User registers an account via the mobile app
2. User powers on device, device broadcasts a wifi signal, User connects to devices wifi
3. User input username, password, wifi ssid, and wifi password at <http://192.168.4.1/>
4. Device stores variable in the firmware, connects to wifi, and obtains the users JWT
5. User takes measurement from the load cell
6. User prints label
7. Device formats measurement and send to API server via HTTP POST
8. Backend stores in Database
9. Database measurements are fetched and displayed on mobile app



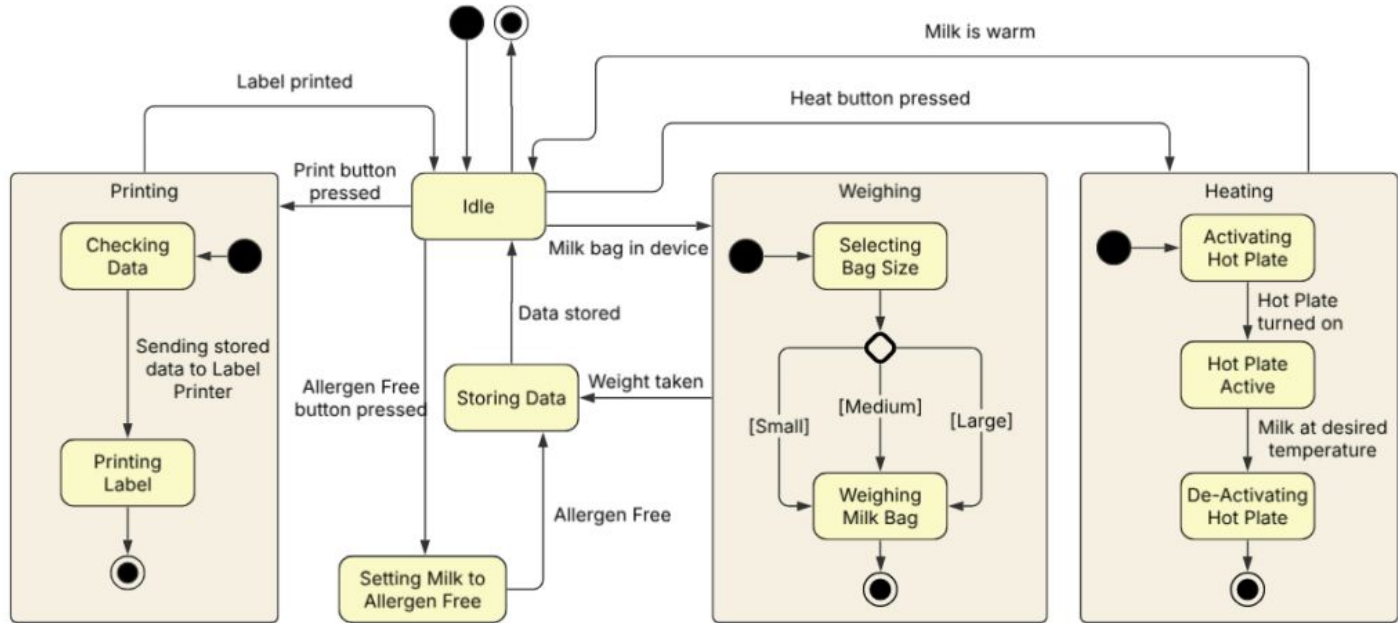
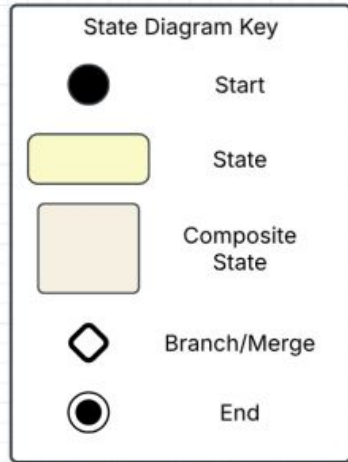
Software Use Case Diagram



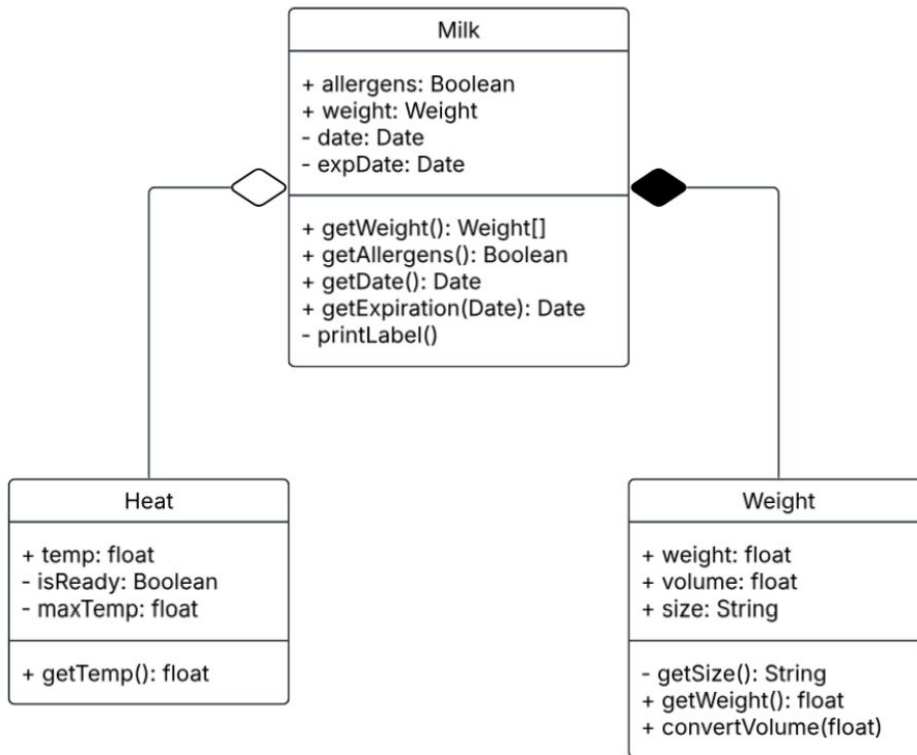
Software Flowchart



Software State Diagram



Software Class Diagram



Temperature Sensor



- Originally used a waterproof DS18B20 digital temperature sensor
- Usable temperature range: -67°F to $+257^{\circ}\text{F}$
- Tested temperature: $\sim 75^{\circ}\text{F}$

Problems:

- DS18B20 sensor activated, but gave a reading of 0°C (32°F)

Solution:

- Switched to the Type-K Thermocouple temperature sensor
- Gave much more accurate temperature readings
- Accurate across room temperature water, hot water, and ice water across 10+ trials.

New Problems:

- Type-K Thermocouple takes longer for a reading than anticipated



Heating Element



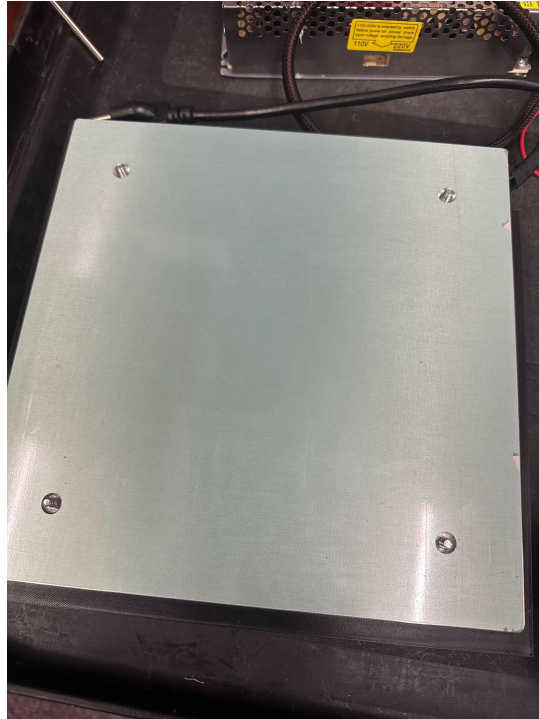
- Dimensions: (235mm x 235mm x 3mm)
- Max Voltage: 24V
- Supplied voltage: 24V
- Uses External PSU

Problems:

- Originally used thermal resistive heating element, was very inconsistent and slow.
- Not very efficient

Solution:

- We switched to a 3D Printer Heating Bed with its own power supply. Uses 24V, can reach 130°F in 10 minutes, and can heat room temperature water and milk bag to 100°F in 20-30 minutes



Trials/Testing for Temperature Sensor & Heating Element



Trial	Thermocouple Temperature (Water)	Thermal Camera Reading	Accuracy
Before Heating	17.8°C (64°F)	16.8°C	94%
5 Minutes	19.7°C (67.5°F)	20°C	98.5%
10 Minutes	25°C (77°F)	25°C	100%
15 Minutes	28.1°C (82.6°F)	28.3°C	99.3%
20 Minutes	33.2°C (91.8°F)	33.5°C	99.1%
25 Minutes	35.7°C (96.3°F)	36°C	99.16%
~27 Minutes	37°C (98.6°F)	37.2°C	99.46%
30 Minutes	37.9°C (100.2°F)	38°C	99.73%
~33 Minutes	39.5°C (103.1°F)	40°C	98.75%
35 Minutes	41.1°C (106°F)	41.4°C	99.27%

Load Cell



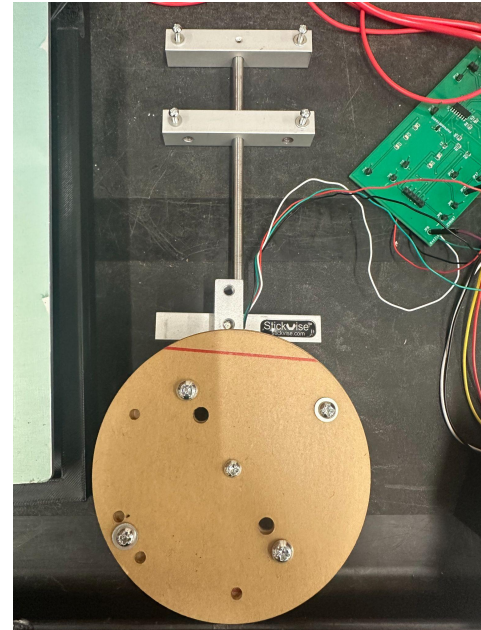
- Max weight: 10 kg
- Tested weight: 100 g - 1000g

Problems:

- Originally gave a reading of about 90 g
- No longer gives accurate readings of the weight
- Doesn't update the changes in the weight applied to the load cell

Solutions:

- Upscaled to a 10 kg load cell
- Brought in a kitchen scale to compare against the load cell readings
- Calibrated load cell effectively using known object weights
- Load cell now shows 99% accurate weight across 10 trials



Trials/Testing for Load Cell



Trial	Item	Known Weight/Volume	Recorded Weight/Volume	Accuracy
1	Water Bottle	500 g (500 mL)	499 g	99.8%
2	Water Bottle	500 g (500 mL)	507 g	98.6%
3	Water Bottle	500 g (500 mL)	499.15 g	99.83%
4	Water Bottle	500 g (500 mL)	498.98 g	99.8%
5	Tea Bottle	500 g (500 mL)	501.75 g	99.65%
6	Tea Bottle	500 g (500 mL)	502.66 g	99.46%
7	Tea Bottle	500 g (500 mL)	501.44 g	99.7%
8	Tea Bottle	500 g (500 mL)	503.9 g	99.2%
9	Milk Bag (in bucket)	~6 fl oz (177.5 mL)	~6.3 fl oz (186.8 mL)	95%
10	Milk Bag (in bucket)	~6 fl oz (177.5 mL)	~6.32 fl oz (187 mL)	94.66%

TTL Thermal Printer

Problems:

- Printer was originally configured using RS232, ESP32 would not read
- Brought in TTL level shifter, they provided inconsistent results
- Print font was light and illegible

Solutions:

- Learned how to configure the thermal printer natively to TTL
- Increasing the amperage to 2.5 allowed the printer to print more clearly
- Printer has been thoroughly tested with more than 25 trials across different measurements, strings and font styles



Trials/Testing for Thermal Printer



Trial of Printer	Preferred Print Time (Seconds)	Time of Print (Seconds)	Display of Print	Accuracy of Display Time
Trial 1	< 5	4.08	Volume, Time, Date, Expiration	100%
Trial 2 (Recorded)	< 5	3.83	Volume, Time, Date, Expiration	100%
Trial 3	< 5	4.23	Volume, Time, Date, Expiration	100%
Trial 4	< 5	4.19	Volume, Time, Date, Expiration	100%
Trial 5 (Recorded)	< 5	4	Volume, Time, Date, Expiration	100%
Trial 6	< 5	4.26	Volume, Time, Date, Expiration	100%
Trial 7	< 5	4.1	Volume, Time, Date, Expiration	100%
Trial 8 (Recorded)	< 5	4.21	Volume, Time, Date, Expiration	100%
Trial 9	< 5	4.17	Volume, Time, Date, Expiration	100%
Trial 10	< 5	4.25	Volume, Time, Date, Expiration	100%

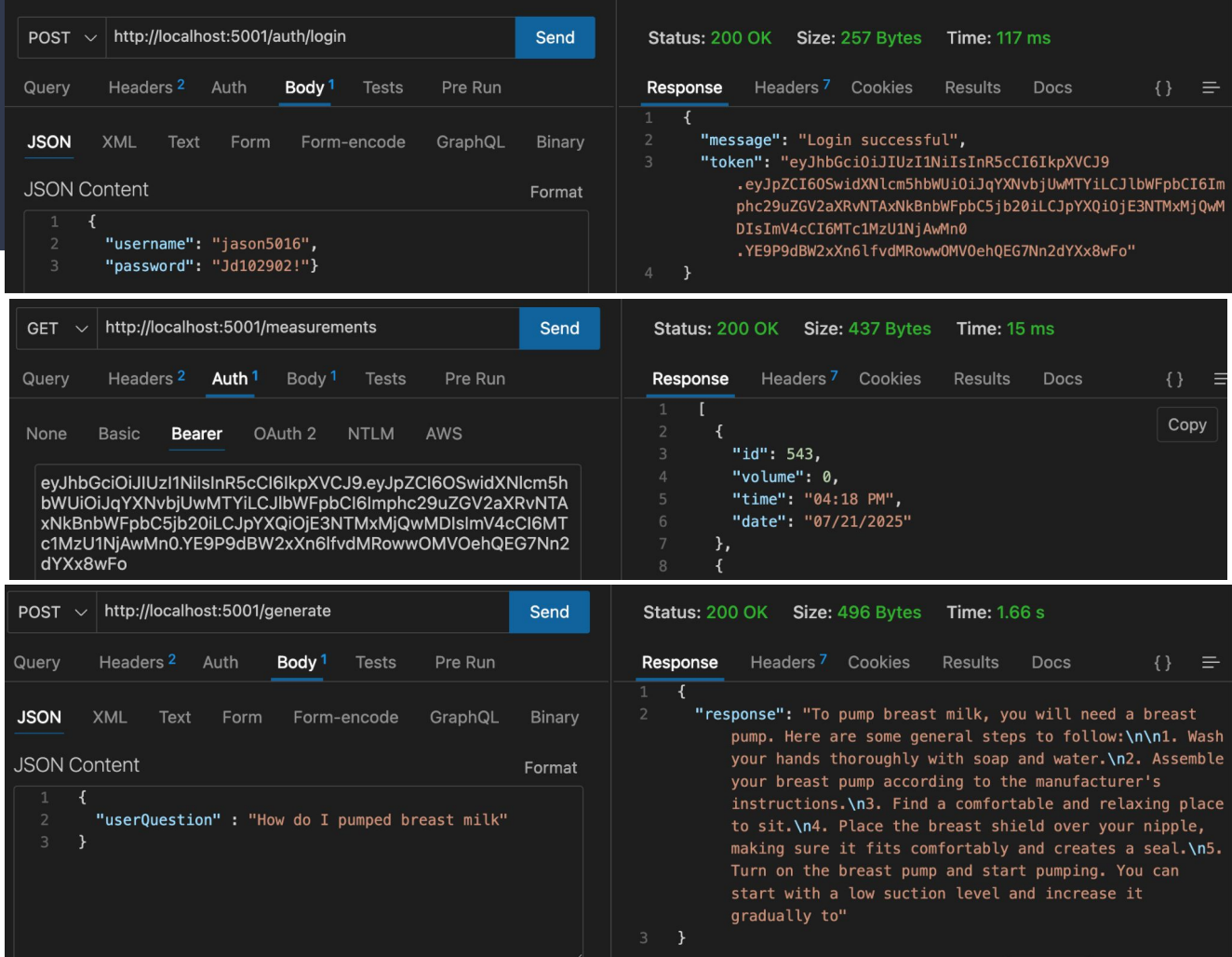
- Had to hardcode user credentials and Wi-Fi information into firmware which limited device flexibility and user experience
- Needed to find a way for measurements to be sent to a specific user

- ESP32 in access point(AP)-mode serves a simple HTML form that any phone can connect to via <http://192.168.4.1/>
- User submits username/password + target Wi-Fi SSID/Pass
- ESP joins that network, send user credentials POST /auth/login → **JWT** is returned and stored to specific user
- ESP32 remembers credentials using Preferences library



Trials include stress testing API endpoints

- POST /auth/login - Testing JWT
- POST /measurements - testing adding measurements with specific JWT
- DELETE/PATCH/GET/measurements - testing with specific JWT
- POST /generate - testing ChatGPT embedded output



Challenges and Solutions



During our design process we ran into a couple of problems. Below are the main issues we ran into and some solutions we are currently working on.

Hardware:

- Heating efficiency of original pads were inconsistent and required more voltage than expected.
 - A solution to this was to incorporate a 3D printer hot bed and a secondary PSU instead. This allowed for a more even distribution of heat while also taking care of the power requirement responsibilities from the MCU
- Weighing the bags of milk proved to be difficult.
 - Switching to a new and larger load cell improved accuracy
- Thermal printer integration has proved to be difficult to integrate.
 - We found a way to switch the printer's native language to TTL instead of RS232, which allowed the ESP32 to properly communicate with the printer

Software:

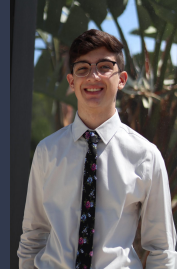
- Wi-Fi “brownout” while using hotspot.
 - The solution to this was to use the “Maximum Compatibility” feature for the iPhone's hotspot

Budget



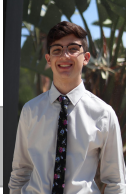
Component	Total Price
Peripheral Components	Load Cell + Display + Printer + Heating + Temperature Sensor + Power Supply, Transformer \$229.78
PCB Components	ESP32 + Resistors + Capacitors + Inductors + ICs + Diodes + Regulators \$259.94
PCB Manufacturing	\$173.60
Enclosure Manufacturing	~\$50
Total	\$713.32

Team Roles and Work Distribution



System/Task	Primary Engineer	Secondary Engineer
Administrative	Zachary Celentano	Jason Devito
Power Supply	Zachary Celentano	Jamieson Foley
Component prototyping & integration	Jamieson Foley/Isaiah Robinson	Zachary Celentano/Jason Devito
Thermodynamic Evaluation	Jamieson Foley	Zachary Celentano
Mobile App Development	Jason Devito	Isaiah Robinson
Backend Development	Jason Devito	Isaiah Robinson
Firmware Programming	Isaiah Robinson	Jason Devito
PCB Design	Zachary Celentano	Jamieson Foley
Enclosure Construction	Jamieson Foley/Zachary Celentano	Isaiah Robinson/Jason Devito

Final Progress



Component	Category	Status	Notes
ESP32-WROOM	Hardware	Integrated	Mounted, powered via 3.3V regulator. Uses USB-USB connection paired with a CP2102
Load Cell and HX711 amplifier	Hardware	Integrated	Reads & displays weight, fully calibrated, mounted and attached to housing
Type-K ThermoCouple and MAX6675	Hardware	Integrated	Ambient, hot and ice-bath tested
3D printer bed heater	Hardware	Integrated	Reaches 100 °F in ~20 min on external power supply
20x4 I2C LCD	Hardware	Integrated	Displays weight, temperature, time, label status and user button controls on 0x27 address
Button Controls	Hardware	Integrated	Implemented for all components
PCB Hardware	Hardware	Integrated	Includes Regulators, Amplifiers, rectifier, & MCU.
Mobile App - Login/Register	Software	Integrated	Active and using API endpoint
Mobile App - Dashboard	Software	Integrated	Active, communicating with server
Mobile app and device connectivity	Software	Integrated	Active, uses wifi, JWT and HTTP post requests

Future Work for Project



- Consolidate PCBs
- Minimize PCB form factor
- Minimize housing form factor
- Minimize peripheral power consumption
- Host backend over reliable fast deployment method
- Refactor Mobile App UI via react native to deploy on both IOS and Android
- Deploy to IOS and Android app stores
- App Stress Testing and optimization

Thank You! – Q & A