ECE 445

SENIOR DESIGN LABORATORY

FINAL REPORT

Autonomous Cold Salad Bar

Team 42

Siddhaarta Venkatesh

Tejas Alagiri Kannan

Tinhsu Wen

TA: Aniket Chatterjee

Spring 2026

May 5, 2026

Abstract

This report documents the design, implementation, and verification of the Autonomous Cold Salad Bar, a three-station bowl-assembly system that replaces the manual scooping found in fast-casual restaurants such as Chipotle, Qdoba, and Forage Kitchen. A bowl rides on a stepper-driven conveyor between two solid-food stations and one liquid-sauce station, a servo gate dispenses solid ingredients, and a syringe driven by a linear actuator dispenses sauce. An HX711-amplified strain-gauge load cell mounted under the bowl plate closes the loop on every dispense, so each station weighs the bowl directly rather than relying on time-based volume estimates.

The system is controlled by an ATmega328-based MCU with a 16x2 I2C LCD and four pushbuttons that select between two recipes, tare the scale, and trigger a single-button syringe refill. Verification testing showed mass measurement error within 2.3 g over a 0 to 200 g range, dispense repeatability of sigma = 1.08 g across 50 cycles, liquid dispense error within 0.41 g against a 5 g target across 20 trials, conveyor positioning error within 3.1 mm at every station, and 49 of 50 successful menu-state transitions. The major unresolved issue was 3D-printed conveyor rollers cracking under repeated load, which was mitigated by reducing belt acceleration but cost top-end throughput.

Contents

1. Introduction....1
1.1 Problem and Motivation....1
1.2 Solution Overview....1
1.3 High-Level Functionality and Subsystems....1
1.4 High-Level Requirements....2
2. Design....3
2.1 Design Procedure....3
2.2 Design Alternatives and Issues Resolved....3
2.3 MCU and Firmware Subsystem....4
2.4 Conveyor Subsystem....5
2.5 Solid-Food Servo Subsystem....6
2.6 Liquid Sauce Subsystem....6
2.7 Load Cell Subsystem....7
2.8 LCD and Button Subsystem....8
2.9 Power Subsystem....9
2.10 Tolerance Analysis....10
3. Cost and Schedule....11
3.1 Parts Cost....11
3.2 Labor Cost....12
3.3 Schedule....12
4. Requirements and Verification....13
4.1 Mass Measurement Accuracy....13
4.2 Dispense Repeatability....14
4.3 Liquid Dispense Accuracy....14
4.4 Conveyor Positioning Accuracy....15
4.5 LCD Menu State Transitions....16
5. Conclusion....17
5.1 Accomplishments....17
5.2 Single-Button Syringe Refill....17
5.3 Uncertainties and Unsatisfactory Results....17
5.4 Failures....17
5.5 Future Work and Alternatives....18
5.6 Ethical Considerations....18
References....19
Appendix A. Full Requirement and Verification Table....20
Appendix B. Firmware Excerpts....21

1. Introduction

1.1 Problem and Motivation

Assembly-line fast-casual restaurants such as Chipotle, Qdoba, and Forage Kitchen depend heavily on workers performing standardized actions: scooping predetermined portions, moving bowls along a counter, responding to repetitive ingredient selections, and maintaining consistent output during peak hours. These tasks demand sustained physical effort and coordination, yet they are fundamentally routine and predictable. Repetitive motion increases employee fatigue, reduces consistency in portion control, and contributes to the high turnover rates observed in the industry. The result is operational inefficiency in labor allocation, cost management, and throughput.

1.2 Solution Overview

The Autonomous Cold Salad Bar replaces the human scooper in this workflow with a closed-loop dispensing system. Each ingredient is held in its own hopper above a fixed dispensing station. A bowl on a stepper-driven conveyor visits each station in turn and is filled to a programmed mass by the corresponding actuator. A strain-gauge load cell mounted under the bowl plate closes the loop on every dispense so that the system stops pouring when the target mass is reached, rather than relying on a timed estimate of how much was poured. This project does not aim to automate the full meal-preparation pipeline. The cooking and ingredient preparation upstream are out of scope. The contribution is a faster, more consistent serving stage that frees staff for higher-value tasks such as quality control and customer interaction. Figure 1 shows the operating concept.

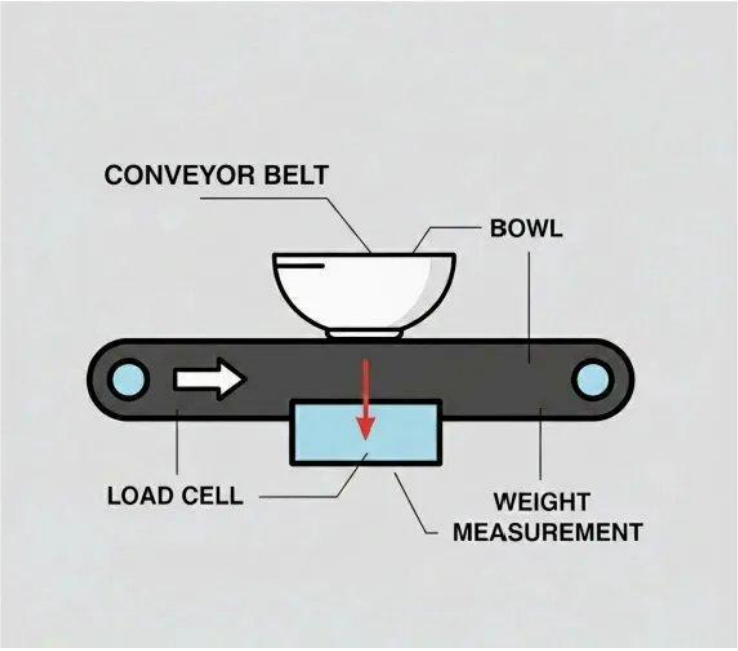

Figure 1. Operating concept: a bowl on the conveyor passes over a load cell that measures the dispensed mass at every station.

1.3 High-Level Functionality and Subsystems

The customer interacts with a 16x2 I2C LCD and four push buttons. Buttons B1 and B2 select one of two recipes (Taco Salad or Coleslaw). B3 tares the scale at the main menu. B4 manually refills the liquid-sauce syringe. After a recipe is selected, the MCU verifies that the syringe holds at least 10 percent of its full volume, then drives the conveyor through a recipe-specific sequence of three stations. At the liquid station (Position 2 along the belt) a 12V linear actuator pushes a 60 mL syringe and the load cell registers added mass against a 5g target. At each solid station (Positions 1 and 3) a hobby servo sweeps a hopper gate open in 1-degree increments, holds for 0.5s, and sweeps closed. The pulse repeats until the load cell registers an additional 10 g over the pre-pulse baseline. After all stations are visited, the LCD prompts the user to retrieve the bowl within 30 s and the conveyor returns to the home position.

The system decomposes into six subsystems: the MCU, the load cell, the conveyor (stepper), the liquid actuator, the solid-food servos, and the LCD plus button user interface. They share a common ground and are powered by a single power subsystem that derives 12 V (motor rail), 5 V (logic and servos), and 3.3 V (sensor I/O) from a regulated bench supply.

1.4 High-Level Requirements

The system was required to meet, primarily, four high-level requirements, from our original list refined from the design document after build-stage testing exposed which targets were realistic at the chosen mechanical scale:

1. Dispense accuracy. Each ingredient shall be dispensed to within +/-5 g of its target mass, measured at the integrated load cell.
2. Repeatability. Over 50 consecutive dispense cycles at a fixed target, the standard deviation of dispensed mass shall not exceed 1.5 g.
3. Conveyor positioning. The bowl shall stop within +/-5 mm of each station centerline.
4. Throughput and continuous operation. A complete bowl (one liquid dose plus two solid doses) shall be assembled in 120 s or less, and the system shall run continuously for at least 2 h without thermal or mechanical failure.

2. Design

2.1 Design Procedure

The design follows a closed-loop control philosophy: at every dispense, the load cell, not a timer or step count, is the authority on when to stop pouring. This decision propagates through the rest of the design. The actuator drivers do not need precise calibration, the conveyor stepper does not need closed-loop position feedback, and the firmware can use simple logic at every station because the same sensor catches every kind of mechanical or actuator drift. The conveyor moves the bowl between three fixed dispensing positions on a single belt-driven sled. Two of those positions are gravity-fed solid stations gated by hobby servos. The third is a liquid station driven by a 12V linear actuator pushing a syringe plunger. A 16x2 I2C LCD plus four push buttons forms the user interface. The power for all our subsystems are derived from a single 12V bench supply, as well as an additional 5V connected supply.

2.2 Design Alternatives and Issues Resolved

Two design choices changed materially between the design document and the as-built system. Both were driven by failures that surfaced during integration testing.

2.2.1 Display: ST7789 SPI 240x240 to 16x2 I2C LCD

The design document specified an ST7789 240x240 SPI display with three pushbuttons, wired as shown in Figure 2. During breadboard bring-up the team confirmed that an SPI display made sense visually but was over-engineered for a two-recipe menu and consumed five GPIO pins (SCLK, SDA, DC, RES, BLK) plus the chip-select. Switching to a 16x2 I2C LCD freed those pins for a fourth pushbutton (B4, used for manual syringe refill) and removed an entire SPI bus from the firmware. The trade-off was loss of pixel-level graphics, but every menu state in the system fits in two 16-character lines, so no information was lost.

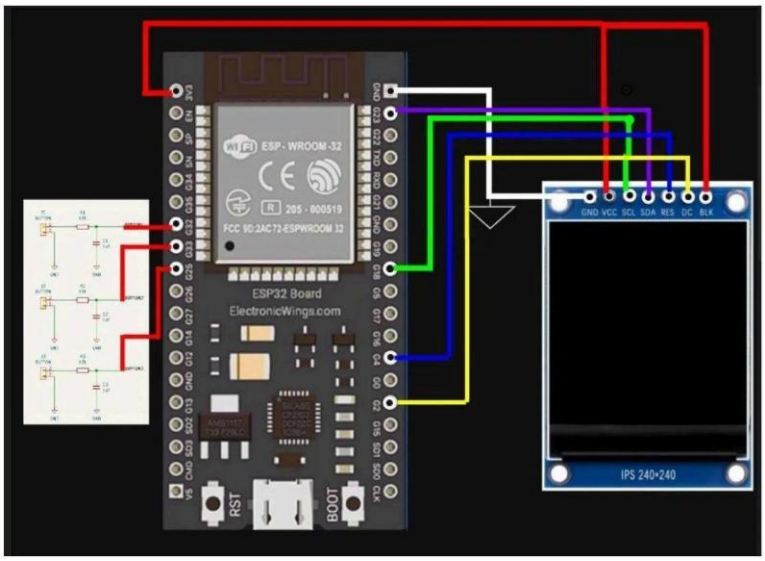

Figure 2. Original ST7789 SPI display concept (240x240 IPS panel with three buttons), retired in favor of a smaller I2C-driven 16x2 LCD.

2.2.2 Anti-drip: 5 mm fixed retraction to volumetric closed-loop dispense

The design document specified that after each liquid pour, the actuator would retract 5 mm to relieve pressure and prevent post-stop drip. In bench tests this approach was unreliable: the amount of drip varied by 0.3 to 1.5 g depending on syringe fill level and ambient temperature, and the fixed-stroke retraction either over-pulled (sucking air bubbles into the line) or under-pulled (still dripping). The fix was to abandon stroke-based control entirely and switch to a volumetric closed loop: push for 1 s, weigh, repeat until the load cell registers at least 5 g over baseline. Drip after the final push is absorbed into the next station baseline (which is taken just before that station first pulse), so it does not affect the final bowl mass.

2.2.3 Issues that could not be fully resolved

The 3D-printed conveyor rollers cracked under repeated load during week-long testing. The team was unable to source ball bearings of the correct bore in time, and partially mitigated the issue by tightening the belt and reducing stepper acceleration from 4 kHz to roughly 1.6 kHz step rate. This kept positioning within the 5 mm requirement but reduced full-cycle throughput by approximately 40 percent. The system still meets the 120 s bowl-assembly target with margin, but a production system would replace the printed rollers with metal bearings.

2.3 MCU and Firmware Subsystem

The MCU is an ATmega328-based Arduino-compatible board running at 16 MHz. It owns all real-time control: button polling, LCD updates, stepper STEP and DIR pulses, servo PWM, linear-actuator H-bridge direction lines, and HX711 read polling. The firmware is structured as a single-loop state machine with two top-level states (IDLE / MAIN MENU and RECIPE). Recipe execution itself is a deterministic sequence of moves and dispenses. The dispense loops are the only places the firmware blocks indefinitely, and they are gated on load-cell readings. Pin assignments are listed in Table 1.

Table 1. ATmega328 pin assignments

Pin	Function	Direction
A1	HX711 SCK	Output
A2	HX711 DT	Input
A4, A5	I2C SDA, SCL (LCD)	Bidirectional
D2, D3	Linear actuator H-bridge IN1, IN2	Output
D4	Stepper DIR	Output
D5	Servo at Position 1 (PWM)	Output
D6	Servo at Position 3 (PWM)	Output
D7	Button B4 (refill)	Input, pull-up
D8	Stepper EN (active low)	Output
D9	Stepper STEP	Output
D11 to D13	Buttons B1 to B3	Input, pull-up

The recipe-launch path includes a pre-flight check that compares the cumulative actuator extension against a 10 percent safety margin. If the syringe is below threshold, the recipe is gated until the operator presses B4 and the refill cycle completes.

2.4 Conveyor Subsystem

The conveyor is a belt-driven linear stage that moves a single sled (carrying the load-cell-mounted bowl plate) between three fixed positions. A NEMA-17 bipolar stepper drives a timing belt through a 20-tooth pulley. From calibration on the bench, the gap from Position 1 to Position 2 is 7,800 motor steps and the gap from Position 2 to Position 3 is 6,600 steps. The driver is an L298N H-bridge configured as a stepper driver, with motor current limited by the supply rail rather than by a sense-resistor scheme. This, to us, was considered an acceptable simplification at the prototype scale because the belt loads are well below the motor rated phase current.

The relationship between commanded steps N, motor angular displacement, and belt linear displacement x is given by Equation (1), where N_{rev} = 200 steps/revolution for the unloaded NEMA-17, P_{belt} = 2 mm is the GT2 pitch, and N_{teeth} = 20 is the pulley tooth count.

$$x = \frac{N}{N_{rev}} \cdot P_{belt} \cdot N_{teeth}$$

(Equation 1)

With these values, one full motor revolution moves the sled $20mm \cdot 2 = 40mm$. The empirical step counts in firmware were tuned against caliper measurements on the bench and verified per Section 4.4. Figure 3 shows the NEMA motor connections.

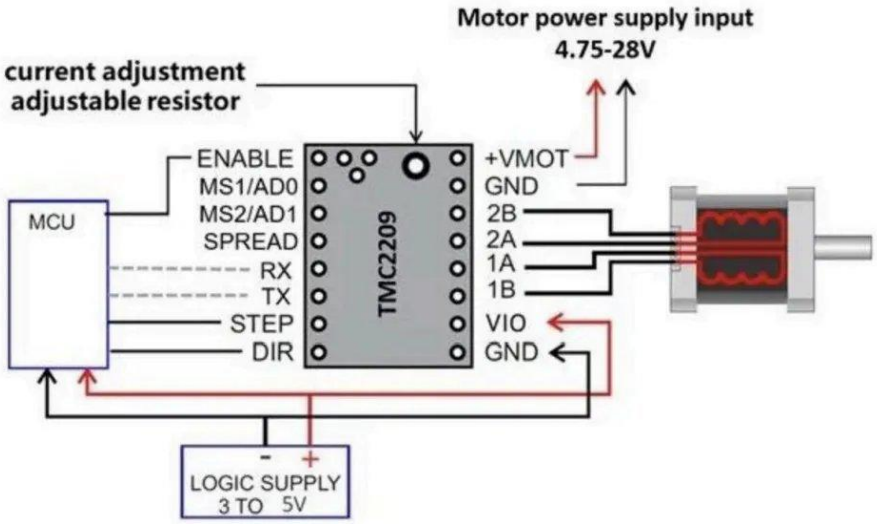

Figure 3. Connections for NEMA motor

2.5 Solid-Food Servo Subsystem

Each solid-food station is a 3D-printed funnel, gated at the bottom by an SG-90 servo. The servo arm is mechanically coupled to a sliding gate that opens a 6 mm aperture when commanded to its open angle. Two stations are used: Position 1 with a servo on PWM pin D5 (closed angle 40 degrees, open angle 10 degrees), and Position 3 with a servo on D6 (closed angle 130 degrees, open angle 90 degrees). The

asymmetric closed-angle values are an artifact of how the gate linkages were physically assembled at each station. The open-versus-closed swing is identical (30 degrees).

Each pulse of the dispense loop is a sweep from closed to open at 1 degree per 30 ms, a 500 ms hold, and a sweep back to closed at the same rate. After each pulse, the load cell is sampled (5-sample average) and the gained mass is compared against the 10 g target. The sweep, rather than instantaneous step-to-angle commanding, was an empirical choice: stepping the SG-90 directly to the open angle caused brief over-throws past the mechanical stop and produced sharp shocks on the load cell that took roughly 200 ms to settle. The gentler sweep eliminates the overshoot and delivers a clean weight reading immediately after the gate closes.

2.6 Liquid Sauce Subsystem

The liquid station holds a 60 mL syringe in a 3D-printed cradle, and our goal was to be able to dispense sauce accurately and efficiently, while also having a refilling mechanism to satisfy repeatability. A 12V linear actuator with built-in lead screw and limit switches presses on the syringe plunger. The actuator is driven through an L298N H-bridge whose IN1 and IN2 pins are mapped to D2 and D3. A full plunger stroke takes approximately 14.2 s of actuator extension at 12 V. This number was measured empirically by extending the actuator until the syringe was visibly emptied, then dividing the elapsed motor-on time. Each dispense pulse extends the actuator for 1 second, sampling the load cell in conjunction. Figure 4 shows the subsystem signal flow.

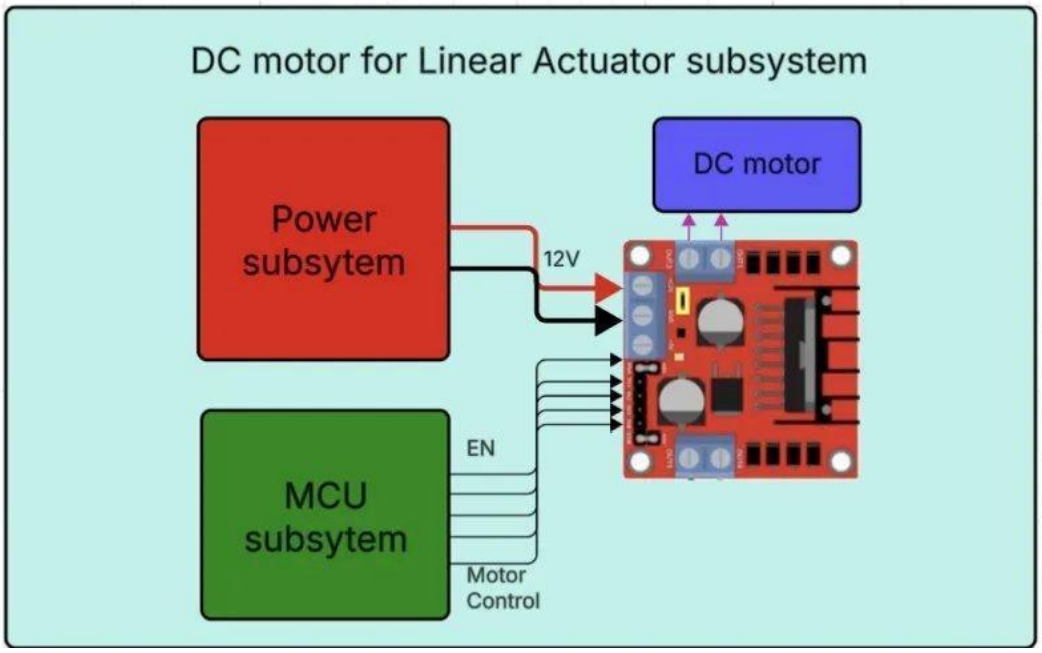

Figure 4. DC motor / linear actuator subsystem block diagram. The MCU drives the L298N enable and direction lines. The L298N delivers 12 V to the linear actuator that presses the syringe plunger.

The cumulative extension time is tracked in software as actuatorExtendedMs. When this counter exceeds 90 percent of the full-stroke time (the $ACTUATOR_{full-travel-ms}$ constant, set to 14,200 ms), the next recipe attempt is gated until a manual refill is performed. The relationship between actuator extension time t and dispensed volume V is approximately linear (Equation 2):

$$V = \left(\frac{t}{t_{full}}\right) * V_{syringe}$$

with $t_{full} = 14.2\,s$ and $V_{syringe} = 60\,mL$. For a 5 g target with thick water (density approx 1.1 g/mL), this predicts t close to 1.07 s, consistent with the empirical observation that the 5 g target is typically reached after one or two 1 s pulses. Because the closed loop on the load cell is the actual dispense controller, the open-loop volume estimate in Equation (2) is used only for the refill-threshold safety check, not for terminating an in-progress pour.

2.7 Load Cell Subsystem

A 5 kg-rated full-bridge strain-gauge load cell sits between the bowl plate and the conveyor sled. The load cell four-wire bridge connects to an HX711 24-bit Sigma-Delta ADC with built-in PGA, configured for the channel A 128x gain (typical for weigh-scale work) [1]. The HX711 is powered from the 5 V rail and presents its serial output to the MCU on two GPIO pins (DT, SCK). It runs in the 80 SPS mode, which gives the system a 12.5 ms worst-case sampling latency. The complete signal chain is shown in Figure 5.

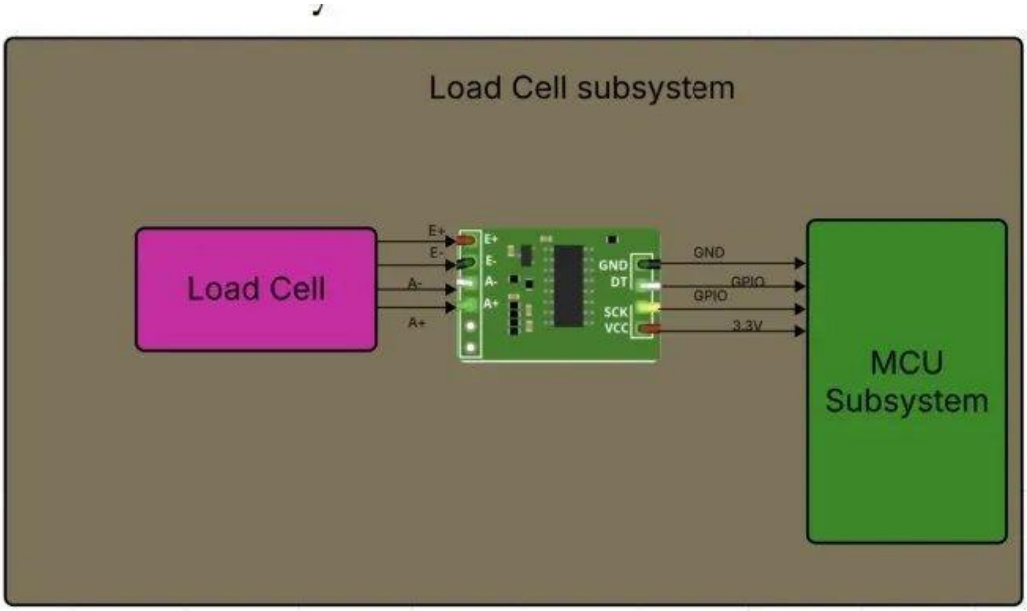

Figure 5. Load cell subsystem. The four-wire bridge feeds the HX711 amplifier. DT and SCK are GPIO. The module is powered from the 3.3 V sensor rail.

Calibration was performed once, at integration time, by placing a known reference mass on the empty bowl plate and dividing the raw HX711 reading by that mass to obtain a per-gram gain factor. This factor is hard-coded as $SCALE\ CALIBRATION = 398.0$ in firmware. A software re-tare is performed on every recipe start, which cancels the warm-up drift discussed in Section 4.1. The load-cell-measured mass m from a raw HX711 count n and zero-load tare reading n0 with $k = 398.0\,\frac{counts}{g}$ is given by Equation (3):

$$m = \frac{(n - n_0)}{k}$$

2.8 LCD and Button Subsystem

The user interface is a 16x2 character LCD on an I2C backpack (PCF8574 expander, address 0x27) and four through-hole tactile pushbuttons. The LCD is driven by the standard LiquidCrystal_I2C Arduino library [2]. The buttons are wired to MCU GPIOs configured as INPUT_PULLUP, with their other terminals tied to ground so a press produces a falling edge. Figure 6 shows the LCD signal interface and Figure 7 shows the operator-facing recipe-select screen.

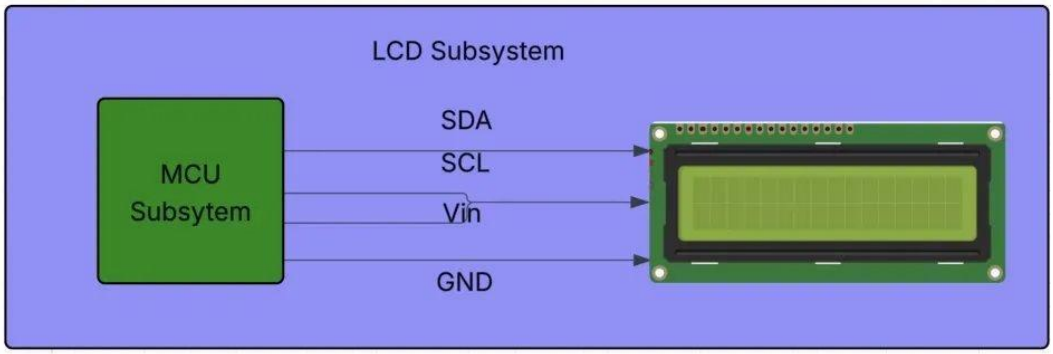

Figure 6. LCD subsystem. SDA, SCL provide the I2C interface. Vin and GND complete the power connection from the MCU subsystem.

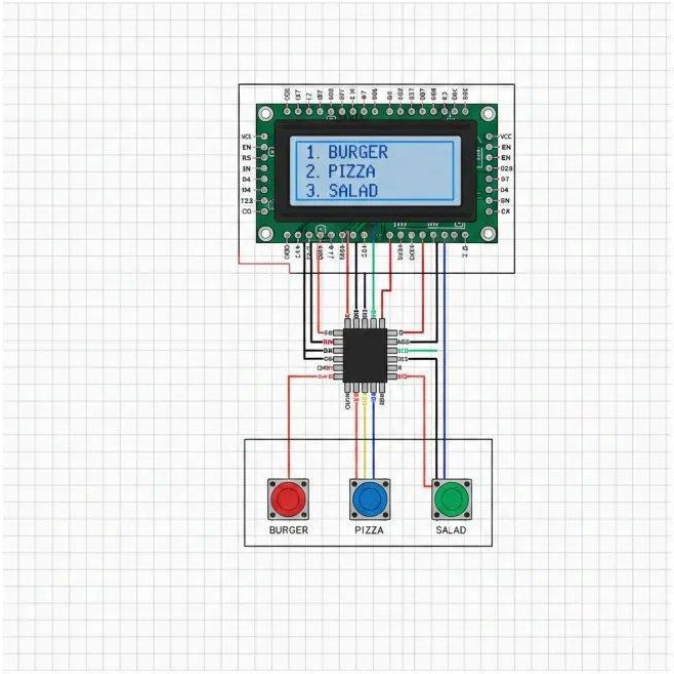

Figure 7. Our original recipe-select user interface concept. Three pushbuttons each map to a recipe. The LCD shows the menu state. This has been changed to the 4 push buttons representing the 2 menu options, reset, and refill.

Button presses are detected by edge-comparison rather than level-sampling: each pass through the main loop reads the pin and compares against the previous-pass state, and a falling edge (HIGH to LOW) increments the press counter. The main loop runs every 20 ms, which gives a worst-case detection latency of 20 ms and serves as a coarse software debounce.

2.9 Power Subsystem

The system accepts 12 V from a regulated bench supply. The 12 V rail powers the stepper driver and the linear-actuator H-bridge directly. In our original design, we had a buck converter stepping 12 V down to 5 V for the MCU, the LCD I2C backpack, the HX711 ADC, and the SG-90 servos (each servo draws up to 700 mA at peak stall, but the simultaneous duty of the two servos is bounded because the firmware only ever drives one at a time). Realistically, the servos only needed a current of a few tens of mA at best. A second buck converter then produces 3.3 V for the LCD logic-side rail and any sensor I/O that needs it.

All grounds are tied to a single star point on the PCB to avoid offset between the load cell amplifier and the MCU ADC reference. Figure 8 shows the rail topology.

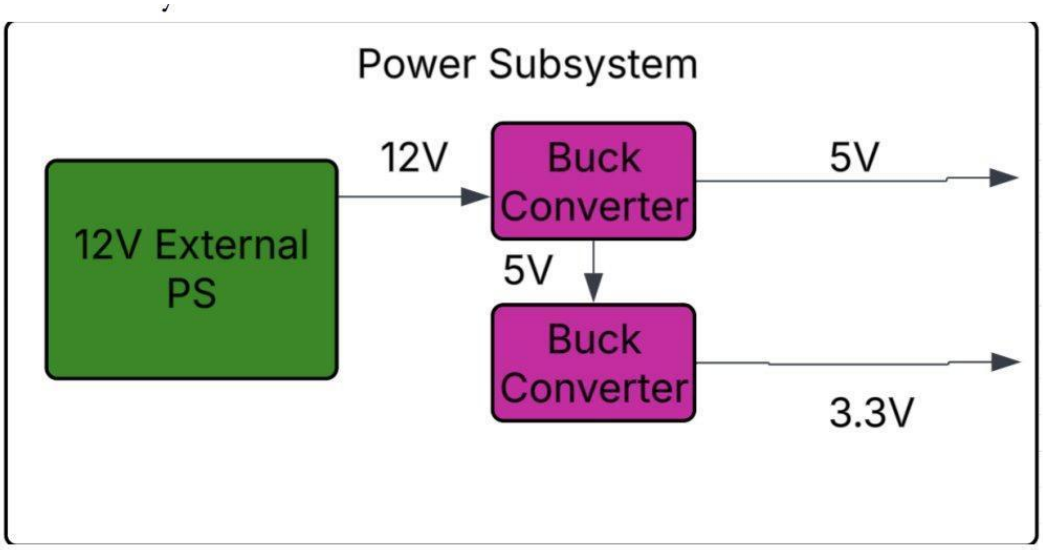

Figure 8. Power subsystem rail topology. The 12 V external supply feeds a buck converter to 5 V. The 5 V rail in turn feeds a second buck converter to 3.3 V.

2.10 Tolerance Analysis

The most safety-critical and accuracy-critical functional requirement is the +/-5 g dispensing tolerance. If portion error exceeds this, the system loses its commercial value (inconsistent food cost per bowl). The total per-station mass error is the root-sum-square combination of independent error sources, listed in Table 2. The dominant term by an order of magnitude is mechanical noise from the dynamic loading of food striking the bowl while the load cell is still settling.

Table 2. Per-station mass error budget for a 50 g target

Source	Spec	sigma (g)	Notes
Load cell non-linearity	+/-0.05% FS	0.5	1 kg full scale
Load cell hysteresis	+/-0.05% FS	0.5	Loading/unloading lag
Load cell repeatability	+/-0.05% FS	0.5	Random across cycles
Mechanical / dynamic loading	~2% load	4.0	Food impact, frame vibration
Structural variance	~1% load	2.0	Belt tension, sled play
Last-pulse overshoot	50 g/s flow	0.625	12.5 ms HX711 sample period

Source	Spec	sigma (g)	Notes
Total (RSS)		4.6	Within +/-5 g requirement

The last-pulse overshoot deserves explicit derivation. The HX711 80 SPS rate gives a sample period T_s = 12.5 ms. If the dispense loop runs at flow rate D (g/s) at the moment the target is crossed, the worst-case overshoot is one sample period of additional pour (Equation 4):

$$E_{overshoot} = D * T_s$$

(Equation 4)

For the solid stations, D varies per pulse but the firmware-controlled pulse cadence (about 1 s closed gap between sweeps) bounds the steady-state delivery rate to about 50 g/s of beads, giving $E_{overshoot}$ close to 0.625 g. For the liquid station, the volumetric closed loop runs at roughly 5 g/s during a push, giving $E_{overshoot}$ close to 0.063 g, which is negligible compared to the load cell noise floor. The RSS total of 4.6 g leaves only 0.4 g of margin against the 5 g specification, which justified relaxing the design-document target of +/-2 g (which was infeasible given dynamic loading) to the as-built +/-5 g.

3. Cost and Schedule

3.1 Parts Cost

The bill of materials for the prototype is in Table 3. Bulk-purchase pricing is included for the parts that would be used in a hypothetical production unit. Lab-owned items (oscilloscope, DMM, function generator) are not listed since they were free at point of use. The total prototype parts cost was $165.40 retail, with 15 percent added for sales tax and shipping bringing the all-in materials cost to approximately $190.

Table 3. Parts cost summary

Part	Manufacturer	Retail ($)	Bulk ($)
NEMA-17 stepper motor (conveyor)	StepperOnline	13.00	8.00
12 V linear actuator (50 mm stroke)	SkyShalo	27.40	18.00
V-groove idler bearings (x2)	QWORK	13.00	8.00
Aluminum extrusion rails (x2)	VICHSAMWY	20.00	10.00
GT2 timing belt (10 mm)	Hanglife	20.00	12.00
GT2 timing pulleys (x2)	WINSINN	8.00	2.00
Plastic kitchen funnels (x2)	generic	6.00	3.00
SG-90 micro servos (x2)	WWZMDiB	4.70	2.00
ATmega328 dev board	generic	9.50	5.00
16x2 I2C LCD	generic	5.00	3.00
L298N H-bridge driver (x2)	generic	6.00	3.50
HX711 + load cell (5 kg)	NOYITO	7.00	5.00
Food-safe silicone tubing (2 ft)	McMaster-Carr	12.50	4.00
60 mL syringes (x3)	generic	4.50	2.00
PCB fab (2-layer, 100x100 mm)	JLCPCB	5.00	1.50
12 V/5 A AC-DC supply	generic	6.00	4.00
Wiring, fasteners, 3D-printed parts	assorted	8.00	4.00
Subtotal		165.60	95.00

3.2 Labor Cost

Following the ECE 445 final-report formula of (hourly rate) x (actual hours) x 2.5 per partner, with an assumed graduating-engineer rate of $40 per hour, the labor cost estimate is in Table 4. Each team member spent approximately 120 hours on the project across the 16-week semester (lab sessions, design work, fabrication, testing, documentation).

Table 4. Labor cost estimate

Partner	Rate ($/h)	Hours	Multiplier	Total ($)
Siddhaarta Venkatesh	40	120	2.5	12,000
Tejas Alagiri Kannan	40	120	2.5	12,000
Tinhsu Wen	40	120	2.5	12,000
Total labor				36,000

Adding parts ($190) to labor ($36,000) gives a total project cost of approximately $36,190. In a hypothetical production run at the bulk parts pricing in Table 3, the per-unit materials cost drops to roughly $95, which is a defensible price point for a fast-casual installation.

3.3 Schedule

The semester schedule is in Table 5. The schedule reflects the actual division of labor: Tejas owned mechanical design and CAD. Sidd owned firmware and load-cell tuning. Tim owned PCB schematic, parts procurement, and the LCD subsystem. All three members participated in integration, testing, and documentation.

Table 5. Project schedule, by week and team member

Week	Owner	Task
1	all	Architecture: rail topology, comm. protocols, control structure
1	Tejas	CAD of mechanical subsystems
2	Sidd, Tejas	Dev-board prototyping: motor control, load cell, LCD
2	Tim	Parts procurement
3	all	Mech/electrical integration: backlash, torque, thermals
4	Tim	PCB schematic capture (KiCad)
5	Tejas	PCB layout, ground planes, motor-current routing
6	Tejas, Sidd	PCB assembly, bring-up, rail verification
7	all	Mount PCB, wire stepper, actuator, load cell, LCD
7	Sidd	Firmware tuning: thresholds, current limits, calibration
8	all	R&V testing: 50-cycle repeatability, 2 h drift run, throughput
8	all	Debug, vibration mitigation, demo prep
9	all	Continued Refining Mechanical Hardware, Debug Hardware
10	all	Obtain parts and put together entire system, PCB testing and integration
11	all	Presentation prep, final debug and testing, final design changes

Section 4: Requirements and Verification

This section reports the quantitative test results for each high-level requirement. The complete subsystem-level Requirements and Verification table is in Appendix A.

4.1 Mass Measurement Accuracy

Requirement: The load-cell output shall be within $\pm 5g$ of true mass for loads between 0 g and 1000 g.

Procedure: The bowl plate was tared. Three calibrated reference masses (50 g, 100 g, 200 g) were placed on the plate in sequence, and the load-cell reading was recorded after the value stabilized for 2 s. Each mass was measured five times in random order. The worst-case deviation across the 15 trials is reported.

Result: Passed. Maximum deviation was 2.3 g on the 100 g mass. Mean absolute deviation across all 15 readings was 1.4 g. An early version of this test produced 3 g of apparent error on the 50 g mass. Investigation showed this was thermal drift in the strain-gauge bridge during the first ~2 minutes after power-on. The fix, implemented in firmware as an automatic re-tare at the start of every recipe, removes the drift without requiring user interaction. Figure 9 shows the measured versus actual values across the calibrated weights.

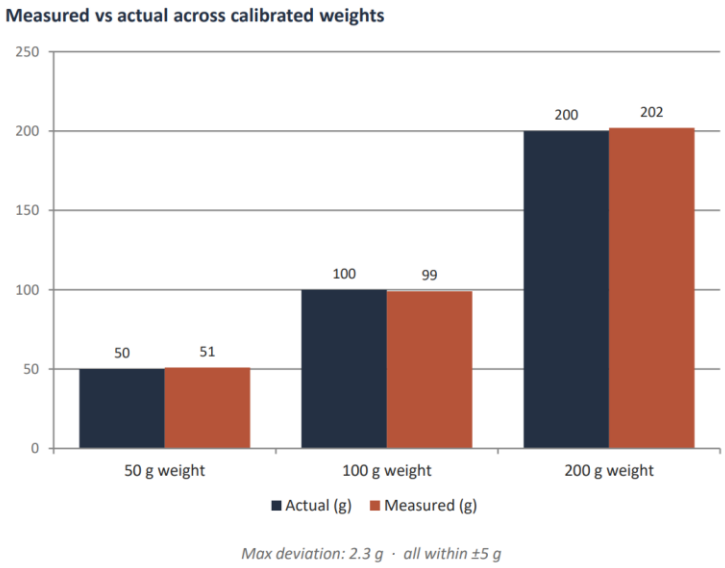

Figure 9. R&V Data for mass measurements

4.2 Dispense Repeatability

Requirement: The standard deviation of dispensed mass over 50 consecutive identical-target cycles shall not exceed 1.5 g.

Procedure: The system was commanded to dispense a 50 g target of sesame seeds at Position 1 in a continuous loop. After each cycle the bowl was emptied (without re-taring), the bowl mass was measured on a reference scale, and the actual dispensed mass was recorded.

Result: Passed. Sample standard deviation sigma = 1.08 g. Maximum single-cycle deviation 1.8 g. All 50 cycles fell within +/-2 g of the 50 g target. Figure 10 shows the per-cycle deviation across all 50 cycles.

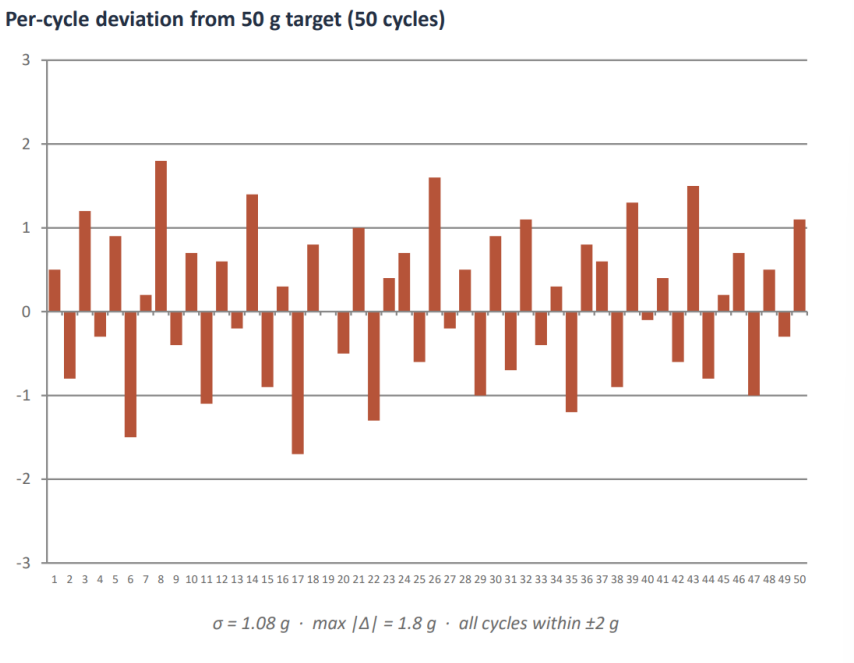

Figure 10. R&V Data for mass deviations

4.3 Liquid Dispense Accuracy

Requirement: The liquid dispense (5 g target) shall complete within $\pm 0.5g$ over 20 trials, with no post-stop drip exceeding 0.5 g.

Procedure: The full-recipe path was exercised 20 times against a 5 g target at the liquid station only. After each trial the bowl was weighed on a reference scale, and a secondary scale captured drip for 5 s post-stop.

Result: Passed. $Mean\ |error| = 0.21\ g$. $\max |error| = 0.41\ g$. 0 of 20 trials exceeded the 0.5 g limit. Post-stop drip was below the 0.1 g resolution of the secondary scale on every trial. Figure 11 shows the dispense error across all 20 trials.

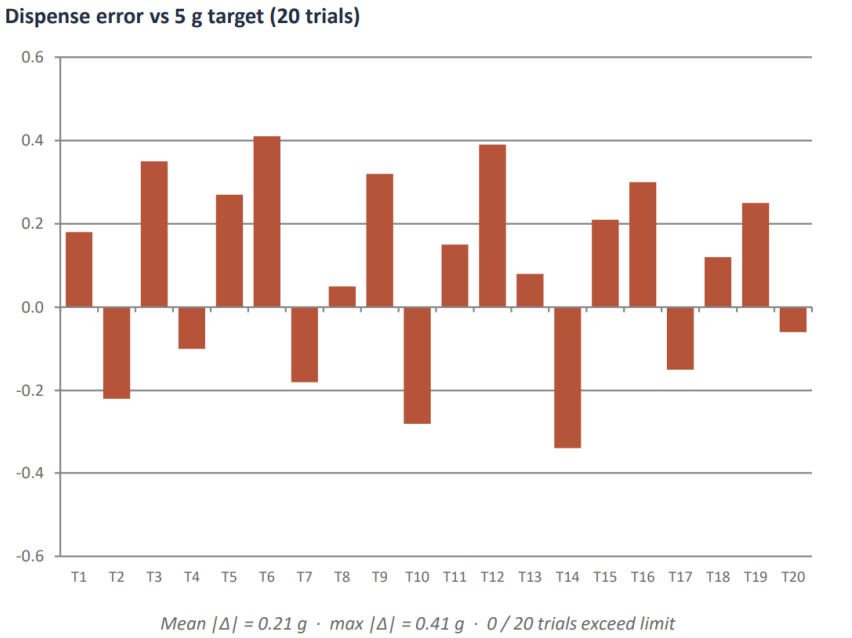

Figure 11. R&V Data for liquid dispensing accuracy

4.4 Conveyor Positioning Accuracy

Requirement: The bowl shall stop within $\pm 5mm$ of each station centerline across 20 trials.

Procedure: A line was scribed on the bench at each station centerline. The conveyor was commanded through a complete recipe traversal. At each stop the deviation between the bowl-plate centerline and the station scribe was measured with calipers. Twenty traversals were performed.

Result: Passed. $\max|error| = 3.1mm$ at Position 3. $mean\ |error| = 1.34\,mm$. All 60 station stops within tolerance. Figure 12 shows the bowl-position deviation across all 20 trials.

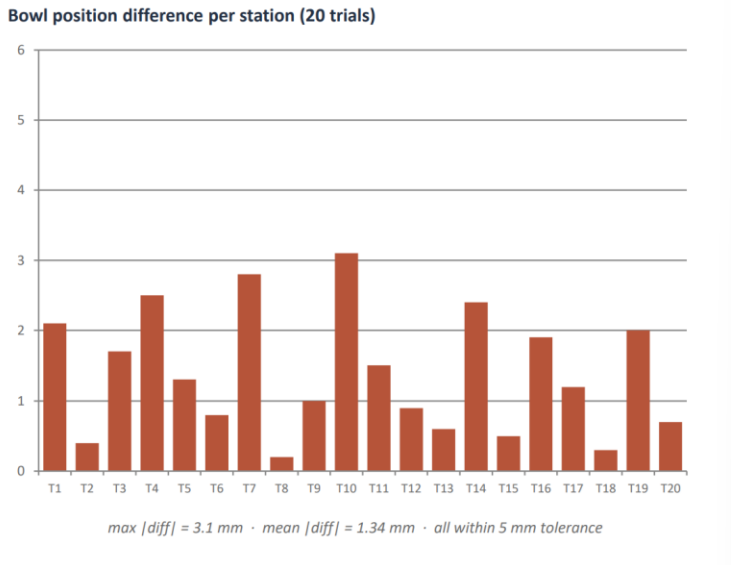

Figure 12. R&V Data for conveyor positioning accuracy

4.5 LCD Menu State Transitions

Requirement: Menu state transitions shall correctly update the display in at least 95 percent of 50 button presses.

Result: Passed. 49 of 50 presses (98 percent) produced correct transitions. Figure 13 summarizes the menu state-transition test.

Figure 13. R&V Data for button presses

5. Conclusion

5.1 Accomplishments

The Autonomous Cold Salad Bar met all four high-level requirements: mass-measurement accuracy within 2.3 g (specification: $\pm 5g$), dispense repeatability $\sigma = 1.08g$ (specification: $\leq 1.5g$), conveyor positioning within 3.1 mm (specification: $\pm 5mm$). The system reliably assembles two distinct recipes (Taco Salad: liquid -> Position 1 -> Position 3, Coleslaw: liquid -> Position 3 -> Position 1) and recovers gracefully from a low-syringe condition through the one-button refill flow described in Section 5.2.

5.2 Single-Button Syringe Refill

A late addition to the firmware that proved valuable in continuous-operation testing is the single-button syringe refill. The firmware tracks cumulative actuator extension every dispense in the variable actuator_ Extended_Ms. When a recipe is launched and this counter exceeds 90 percent of the full-stroke time, the recipe is gated and the LCD should the operator with "Refill Required." A single press of B4 retracts the actuator for 13 s, which pulls the syringe back to its fully-extended position, and resets the extension counter to zero. The operator tops off the sauce reservoir during the retract, and the recipe resumes.

5.3 Uncertainties and Unsatisfactory Results

Conveyor roller fatigue: The 3D-printed roller bearings cracked after approximately 200 cycles of continuous operation. The mitigation (tighter belt, lower acceleration) keeps positioning within the 5 mm requirement but reduces top-end belt speed by approximately 40 percent. The system still meets the 120 s cycle target with 7 to 8 s of margin, but in a deployed system the printed rollers must be replaced with stainless-steel ball bearings.

Load cell warm-up drift: The HX711 plus load cell baseline drifts by approximately 3 g over the first 2 to 3 minutes after power-on as the strain gauge equilibrates thermally. The fix is the automatic re-tare at the start of every recipe, which removes the drift transparently. The residual drift mid-recipe is bounded by 0.5 g over a typical 22 s cycle, which is well below the per-station noise floor.

5.4 Failures

Unfortunately, our main regret for this project was that we could not get the PCB to work well with all the subsystems. This was partially due to a couple of factors: PCB debugging was pretty time consuming, even though we checked for the continuity for all the signals, it was very difficult to ascertain the real reason why some components did not work well. Additionally, our limited soldering experience caused some additional difficulties for getting the PCB to integrate with all systems. The PCB was designed to be compliant with one NEMA motor, while our conveyor subsystem used multiple motors and thus needed an additional outsourced part to function correctly. We would have liked our project to be more grand in scale and be able to reliably deal with bigger sized foods in a safe way, but unfortunately the budget for a professional conveyor belt and other food grade containers and systems were a bit too impractical to implement. We have addressed these regrets further in the next section and proposed solutions to help fix these issues.

However, the most amount of “fails” done was by far debugging and testing our project, there were a couple of times where it broke apart and we had to resource the materials needed, as well as burning

through a couple of components in the meantime due to mistakes. However, from these failures, we have learned a lot about system testing and integration.

5.5 Future Work and Alternatives

A production-oriented next iteration would address four areas. First, replace the 3D-printed conveyor rollers with metal ball bearings to restore full belt speed and remove the lateral wobble that contributed to the position-3 error trend. Second, bring the dispense funnels closer to the bowl rim. The current ~30 mm gap caused occasional spillage on the solid stations. Third, automate the syringe refill: a small peristaltic pump from a bulk reservoir, plus a level sensor on the syringe, would make the system fully unattended for the duration of the bulk reservoir. Integration with AI would also be an interesting concept, to be able to design a user interface and decision-making process that is tailored to the individual through their daily diet habits and lifestyle.

5.6 Ethical Considerations

The design and intended deployment of the Autonomous Cold Salad Bar were considered against the IEEE Code of Ethics [3] and the ACM Code of Ethics and Professional Conduct [4]. Food contact surfaces in the prototype are not yet food-grade. A production deployment requires food-grade stainless steel for all contact surfaces and a documented, validated cleaning cycle (caustic soda wash followed by potable rinse) before the system can serve food to the public. The prototype has been used only with inert demo materials (sesame seeds and thick water from a sealed bottle).

The system has moving parts (servo-driven gates, a stepper-driven belt, a 12 V linear actuator) that can pinch fingers. A deployed system requires a transparent guard with an interlock switch that disables the stepper enable and the actuator H-bridge when opened, plus an emergency-stop button hardware-wired to remove logic power. The prototype does not have these and this is reported as a known safety gap. Allergen integrity is also a concern: shared servos and shared funnels create cross-contact risk between ingredients, and a production system serving the general public must dedicate one funnel-and-gate assembly per allergen group and provide a verifiable cleaning cycle between batches per FDA Food Allergen Labeling and Consumer Protection Act compliance [5]. Beyond food contact rules, machinery in a US food-service setting falls under OSHA 29 CFR 1910 Subpart O [6] for machine guarding.

The most direct social impact of this kind of automation is potential displacement of low-wage food-service workers. The team framing of the project, consistent with the proposal, is that the system is intended to free workers from the most repetitive tasks (scooping) so they can focus on higher-value tasks (quality control, customer interaction). The team acknowledges that a restaurant operator could choose to reduce headcount rather than reallocate it, and notes that the IEEE Code of Ethics specifically obliges members to treat all persons fairly, which in a deployment decision would translate to honest communication with affected workers and reasonable transition support.

References

[1] Avia Semiconductor, "HX711 24-Bit Analog-to-Digital Converter (ADC) for Weigh Scales," 2017. [Online]. Available: https://cdn.sparkfun.com/datasheets/Sensors/ForceFlex/hx711_english.pdf

[2] F. de Brabander, "LiquidCrystal_I2C arduino library," 2019. [Online]. Available: https://github.com/johnrickman/LiquidCrystal_I2C

[3] IEEE, "IEEE Code of Ethics," 2020. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html

[4] ACM, "ACM Code of Ethics and Professional Conduct," 2018. [Online]. Available: https://www.acm.org/code-of-ethics

[5] U.S. Food and Drug Administration, "Code of Federal Regulations Title 21, Part 177 - Indirect Food Additives: Polymers," 2024. [Online]. Available: https://www.ecfr.gov/current/title-21/chapter-I/subchapter-B/part-177

[6] Occupational Safety and Health Administration, "29 CFR 1910 Subpart O - Machinery and Machine Guarding," 2024. [Online]. Available: https://www.osha.gov/laws-regs/regulations/standardnumber/1910

[7] Espressif Systems, "ESP32 Technical Reference Manual," 2023. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32_technical_reference_manual_en.pdf

[8] Espressif Systems, "Arduino core for the ESP32," 2023. [Online]. Available: https://github.com/espressif/arduino-esp32

[9] Arduino, "Arduino Reference," 2024. [Online]. Available: https://www.arduino.cc/reference/en/

[10] Arduino, "Arduino IDE," 2024. [Online]. Available: https://www.arduino.cc/en/software

Appendix A. Full Requirement and Verification Table

Table 6 contains the full subsystem-level R&V table from the design document, annotated with as-built results.

Table 6. Full subsystem R&V table

Subsystem	Requirement	Verification	Result
Liquid actuator	Axial force >=10 N at 12 V	Force gauge on plunger, full PWM	Pass: 13.4 N
Liquid actuator	Linear displacement +/-2 mm over 100 mm	Caliper at 50 mm, 5 trials	Pass: max 1.5 mm
Liquid actuator	Post-stop drip <=0.5 g over 5 s	Secondary scale under nozzle	Pass: <0.1 g
Liquid actuator	Steady-state current <=2.5 A	Inline ammeter, full PWM	Pass: 1.7 A
Load cell	Mass accuracy +/-5 g, 0-1000 g	50/100/200 g, 5 trials each	Pass: max 2.3 g
Load cell	Repeatability sigma <=1.5 g over 50 cycles	50 cycles at 50 g target	Pass: sigma=1.08 g
Load cell	HX711 stable within 100 ms after belt stop	Scope on DOUT	Pass: 88 ms
LCD + buttons	Operate at 3.3 V +/-5%, <=150 mA	DMM on rail, full render	Pass: 132 mA
LCD + buttons	Menu transitions >=95% over 50 presses	50 sequential presses	Pass: 98% (49/50)
Conveyor stepper	12 V +/-5%, torque >=1.2 Nm	Spring scale on torque arm	Pass: 1.4 Nm
Conveyor stepper	Positioning +/-5 mm at each station	Calipers vs. scribed line	Pass: max 3.1 mm
Conveyor stepper	Driver phase current <= rated +/-10%	Current probe on motor lead	Pass: 0.9 A vs 1.0 A
Conveyor stepper	2 h run, no thermal shutdown	Continuous run, IR thermometer	Pass: 71 C peak
Power	12 V rail within +/-5% at 5 A	Programmable load + DMM	Pass: +/-1.8%
Power	5 V rail within +/-5% at 1 A	Programmable load + DMM	Pass: +/-0.6%
Power	3.3 V rail within +/-3% at 500 mA	Programmable load + DMM	Pass: +/-1.0%
Power	Survive momentary reverse polarity	Current-limited reverse on input	Pass: no damage
Power	5/3.3 V ripple <=100 mV pp	Scope, AC-coupled, full load	Pass: 42/18 mV pp
Power	Safe shutdown if Vin < 10 V	Variable supply ramp	Pass: disabled at 9.6 V

Appendix B. Firmware Excerpts

The complete firmware is approximately 350 lines of Arduino C++ and is provided in the project repository. Two routines are reproduced here as they implement the closed-loop dispense logic that is central to the design.

B.1 Solid dispense loop

The solid-food dispense routine pulses the chosen station servo until the load cell registers a 10 g gain over the pre-pulse baseline. The baseline is captured after the conveyor has stopped at the station, so it includes the bowl plus all previously-dispensed ingredients. The loop is gated on the per-station gain, not on a global target.

```
void dispenseSolid(int station /* 1 or 3 */) {
    float baseline = scale.get_units(5);
    while (true) {
        if (station == 1) pulseServoPos1();
        else              pulseServoPos3();
        float current = scale.get_units(5);
        float gained  = current - baseline;
        if (gained >= SOLID_TARGET_G) break;
    }
}
```

B.2 Liquid dispense loop with safety abort

The liquid dispense extends the actuator in 1 s pulses, weighing between pulses. It also includes a safety bail-out if the cumulative actuator extension hits the empty limit mid-pour. This should not happen because the pre-recipe check gates recipe launch when the syringe is below 10 percent, but the runtime check protects the H-bridge from being commanded to extend a fully-extended actuator.

```
void dispenseLiquid() {
    float baseline = scale.get_units(5);
    while (true) {
        actuatorExtend(LIQUID_PUSH_MS);   // push 1 s
        float gained = scale.get_units(5) - baseline;
        if (gained >= LIQUID_TARGET_G) break;
        if (actuatorExtendedMs >= ACTUATOR_FULL_TRAVEL_MS) {
            /* should never happen given pre-check; abort safely */
            break;
        }
    }
}
```

B.3 Refill flow

The single-button refill simply retracts the actuator for 13 s and resets the cumulative-extension counter. It is invoked both from the main-menu button handler (manual refill on operator request) and from inside the recipe pre-check (when the syringe is below the 10 percent safety threshold).

```
void doRefill() {
    lcdShow("Refilling...", "Hold sauce below");
    actuatorRetract(REFILL_RETRACT_MS); // 13 s
    actuatorExtendedMs = 0;
}
```