Weightb0ard Design Document

Team - Thomas Driscoll, August Gress, Kyle Patel
ECE 445 Project Proposal - Spring 2020
Group 8
TA: Dhruv Mathur

1.1 Objective

In any environment that involves cooking or food preparation, knowing the amount of ingredients on hand is of the utmost importance. This can range from large-scale restaurants that have massive quantities of any given food to athletes engaging in meal prep, often down to the gram. In between these two extremes also exist the average consumer, who rely on unreliable memory and insatiable hunger when shopping instead of their objective needs.

Our solution is an internet connected, weight-sensitive kitchen cabinet/tray that pings a grocery list app. For items such as rice, sugar, flour, protein powder, creatine, etc., a scale could measure the amount at home. If it falls below an ingredient-appropriate threshold, a microcontroller will send an update to a user's phone. Simply checking the app once in the store, or while placing a large order, allows the user to purchase the correct amount of food. It will have 7 separately sized sensors that accurately measure ingredient amounts placed on top of it, which will be a proof-of-concept to show our idea's scalability.

1.2 Background

Kitchen preparation is a part of everyday life for many people. From creating meals for themselves or their families to working in a high pressure industrial kitchen that cranks out pounds and pounds of food an hour, there is a constant need to know which ingredients are on hand.

In the pursuit of not running out of ingredients; however, there comes a tendency to overbuy ingredients, generating food waste. Food waste is a global concern, and is the subject of many different studies and research articles on its effects on the environment and society [10]. In addition to overbuying, a business or individual can also forget a particular ingredient that they needed to make a recipe simply because they didn't know they had ran out of it. One potential solution that has been brought to market is the Samsung Family Hub Smart Refrigerator, which has a 3 camera array on the inside of the fridge to view its contents from wherever you are [CITE: https://www.samsung.com/us/explore/family-hub-refrigerator/overview/]. One large issue with this however, is that it is difficult to quantify how much is left of an ingredient as the camera is mounted where you only see the outside packaging of each ingredient. This is something our board proposes to solve by individually weighing each ingredient.

We saw a need to create a way to simplify the lives of everyone who relies on what ingredients they have on hand and prevent overbuying on ingredients that they didn't need. By creating this board, we aim to simplify the lives of those who depend on having the correct ingredients.

1.3 High Level Requirements

- Able to measure the weight of ingredients placed on pressure sensitive pads within ±10% of true weight.
- Board will send information about the ingredients on it to an internet connected database.

- Weight (in grams) is reported in a mobile application using the data stored in that internet connected database

2 Design

2.1 Block Diagram

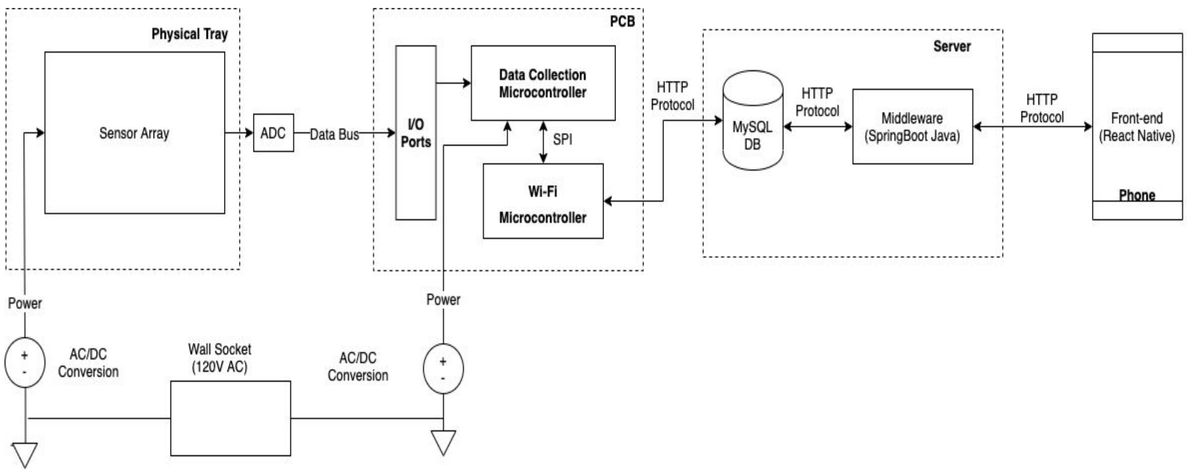

Fig. 1: Block Diagram

2.1.1 Physical Tray:

The physical tray holds the sensor array that will be processing the real-world input of each ingredient. 4 sensors rated to hold a maximum of 100g, 2 sensors rated for a maximum of 500g, and 1 sensor rated for a maximum of 5kg will each hold a platform on the tray. Each sensor has a data path that it sends the current detected weight along, connecting to the I/O ports of the PCB. The sensors are all powered by a wall outlet, represented by the above voltage source diagram (which will convert from AC to DC via a consumer rectifier device).

2.1.2 PCB:

Our PCB will consist of 3 main entities: I/O ports, the microcontroller(Atmega328), and the Wi-Fi adapter (ESP8266EX). The seven I/O ports, one for each sensor, will connect the data path of the sensors to the microcontroller. The microcontroller will handle processing the weights in real time and any other circuit controls necessary. If the microcontroller detects a significant weight change, it will send a HTTP message via the Wi-Fi adapter to our remote database. This ensures that the board and the user can communicate via any common Internet connection.

2.1.3 Linux Server:

The server will be a remote Linux environment that handles the database and incoming/outgoing requests for information. As the weight changes on the board, the database will be updated to reflect the new associated weight for each sensor. When the user makes a request for the current weight, the database will interact with the middleware (written in SpringBoot Java) to send the weights and other pertinent information to the user. The middleware will operate by sending and receiving GET/POST operations following standard HTTP methods.

2.1.4 User Phone:

In order to interact with the board, each user will load an app onto their phone. This app will be the user-interface that allows the user to see what items they need while mobile. Upon opening the app, the user's phone will automatically make a request to the Linux server, where our middleware will direct the appropriate operations to perform. Our plan is to write this front-end in React Native to allow cross-platform support and fast development time.

2.2 Physical Design

Fig. 2 is what our physical design is going to look like for this project from the top down view. Our choices are such that each progressively larger sized sensor has a larger footprint on the board for measuring ingredients. This choice was made so that our board had space to accommodate the larger, heavier ingredients on the large pressure sensors, and the smaller, less massive ones weren't given an unnecessarily large space on the board. All of the S1, S2, S3, S4 sensors will be the same size, as will the M1 and M2 sensors. Each load cell (weight sensor) will be mounted from the edge of the pad, with a platform centered at the other end of the pad, such that the rubber center of the cell can bend to produce sensor outputs.

Fig. 3 is what our physical design is going to look like for this project from the side view perspective. We also wanted to design the board such that there was a moderately sized cavity underneath to house the PCB and wires that came off of the pressure sensors.

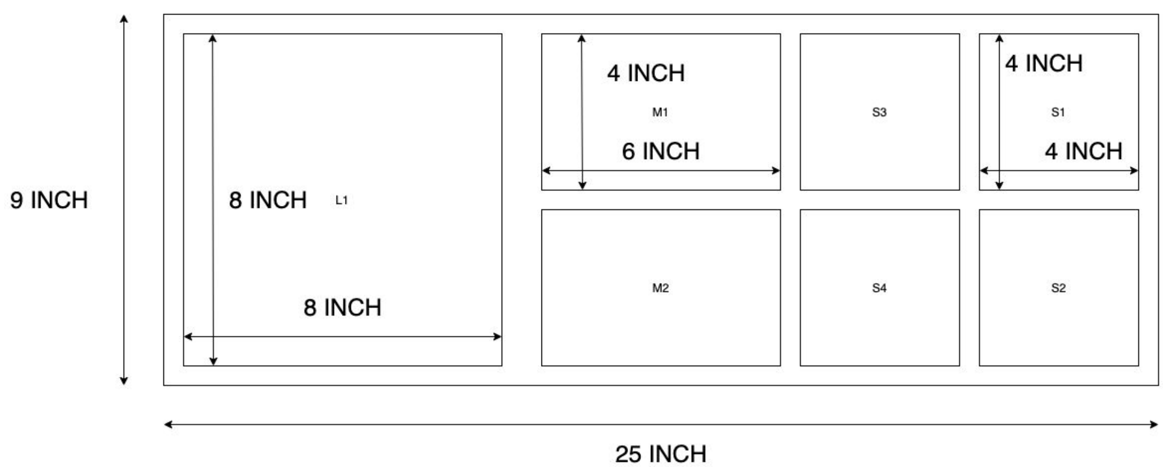

Fig 2. Physical Diagram (Top Down View)

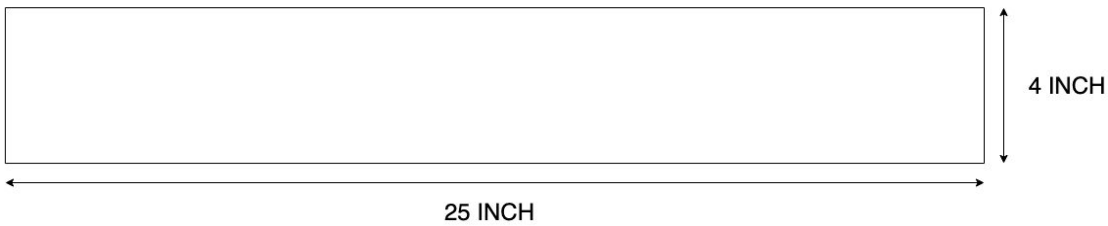

Fig 3. Physical Diagram (Side View)

2.3.1 Physical Tray Subsystem R&V Table

Module	Requirements	Verification
Physical Tray: 7x HX711 Breakout Boards for Load Sensors	1. Each breakout board must receive 5 ± 0.3V by way of linear voltage regulator [1] 2. Each breakout board must receive current of 20 ± 5mA [1]	1. a. Send a 5V DC signal to a breakout board’s EXC(+) pin with all inputs of a single load cell connected. b. Place a small known weight on measure potential difference between EXC(+) and EXC(-) using DMM. 2. a. Send a 5V DC signal to a breakout board’s EXC(+) pin with all inputs of a single load cell connected. b. Measure current across EXC(+) and EXC(-) them using DMM, verify it is within range 15mA-25mA.
Physical Tray: 4x TAL221 100g Load Sensors, 2x TAL221 500g Load Sensors, 1x TAL220B 5kg Load Sensor	1. Each sensor must be mounted in a structurally stable manner such that they can hold plastic containers, a mounted platform, and 100g/500g/5kg of an ingredient 2. Each sensor must be provided with 5 ± 0.3V supply [2][3] 3. Sensors must provide readings within 10% accuracy	1. 2. 3.

PCB Subsystem R&V Table

Module	Requirements	Verification
PCB: Microprocessor (ATMega328P)	1. Must be able to facilitate sequential collection of quantized sensor data at 80SPS (takes ~4-6 samples at 80SPS to	1. 2.

	generate stable reading). [1] 2. Must be able to send digital sensor readings to Wi-Fi Microchip via SPI (serial port)	
PCB: Wi-Fi Microchip (ESP-8266)	1. Latency of transmission of sensor data array to MySQL DB must be no greater than 30 seconds 2. Must store ~200B worth of sensor measurement values via SPI in flash memory(4-6 readings * 7 sensors * 4B / float value)	1. 2.

Server/Phone Subsystem R&V Table

Module	**Requirement**	**Verification**
Server: MySQL DB	1. Database is reachable from a remote address 2. Database supports a table for each sensor 3. Database only accepts input from a recognized user 4. Database inserts data correctly from .json file with timestamp	1. 2. 3. 4.
Server: SpringBoot Java Middleware	1. Accepts GET request for latest weight measurement and returns an HTTP 200 header with JSON data to user-facing application 2. Accepts POST request for latest weight measurement and returns an HTTP 200 ACK to Wi-Fi chip on tray 3. Checks that request is from an	1. 2. 3. 4.

	accepted origin 4. >80% unit test coverage for every HTTP method	
Mobile Application: React Native Front-End	1. Users should be able to view weight information from all sensors in readable format 2. Application should successfully notify user if any readings are below user-defined thresholds for each ingredient 3. Application allows customizable labels for each sensor 4. >80% unit test coverage for every React component	1. 2. 3. 4.

3 Cost and Schedule

3.1.1 Manpower Cost

The average salary of a 2017-2018 ECE Illinois Computer Engineering Grad (as our group is comprised of) was $92,430 [https://ecs.engineering.illinois.edu/files/2019/03/IlliniSuccess_AnnualReport_2017-2018_FINAL.pdf]. Working 52, 40 hour weeks (for a total of 2,080 hours a year), this breaks down to $44.43 per hour. We will use this as our fixed hourly cost for our project. We assume working 10 hours per week, and for the remaining ~10 weeks for this class. This formula also neglects any time working with product marketing or external partnerships. We therefore calculate our manpower cost for this project as follows:

$$2.5 \times Number\ of\ Group\ Members \times Fixed\ Hourly\ Cost \times \#\ of\ Hours\ per\ Week \times \#\ of\ Weeks = Cost$$

$$2.5 \times 3 \times \$44.43 \times 10 \times 10 = \$33,322.50$$

As shown above, we calculate the manpower cost to be $33,322.50 for this prototype.

3.1.2 Part Cost

Part	Cost (bulk)	Cost (prototype)
5x TAL221 100g Load Sensors	$2.80 * 5 = $14	$8.95 * 5 = $44.75
3x TAL221 500g Load Sensors	$2.50 * 3 = $7.50	$9.95 * 3 = $29.85
2x TAL220B 5kg Load Sensor	$1.80 * 2 = $3.60	$10.95 * 2 = $21.90
1x ESP-8266 Wi-Fi Microchip	$1.40 * 1 = $1.40	$6.95 * 1 = $6.95
8x HX711 Breakout Boards	$1.47 * 8 = $11.76	$9.95 * 8 = $79.60
1x ATMega328P Microprocessor	$3.16 * 1 = $3.16	$4.30 * 1 = $4.30
1x TOL-15664 5V 2A AC/DC Wall Adapter	$3.00 * 1 = $3.00	$10.95 * 1 = $10.95
1x Domain Name for Web Hosting	$15.66	$31.32
Misc. Construction Costs (Wood, surface finishing, screws, etc.)	$10.00	$20.00
Final Part Cost		

As shown above, we calculate the bulk cost for our board's parts to be {INSERT} and the prototype cost to be {INSERT}. Accounting for the cost of shop labor hours, we find the total part and shop cost to create our prototype would be {INSERT SHOP COST}.

3.1.3 Total Cost

We calculate the total cost, including manpower, shop labor, and parts, to be {INSERT TOTAL OVERALL COST HERE}.

3.2 Schedule of Work

Week of	Kyle	August	Thomas

2/23	Finish Design Doc	Finish Design Doc, Design PCB after approved	Finish Design Doc
3/1	Complete Soldering Assignment	Engage with ECE Shop, Complete soldering assignment	Set up server and connect to domain name (must be pingable)
3/8	Finalize PCB design for early bird	Place order for PCBWay	Set up SpringBoot Java API and test connection using Arduino Uno
3/15	Program ATMEGA and WIFI chips	Program ATMEGA and WIFI chips on personal testing board	Write unit tests for API, fix any bugs, start front-end dev
3/22	Work on individual progress report.	Work on individual progress report, test completed PCBs	Work on individual progress report, continue front-end dev
3/29		Work on integration between board and mobile application	Work with August on board/mobile app integration, continue front-end dev
4/5		Continue bug-fixing integration locally	Write unit tests for front-end Dev, finish bug testing
4/12	Prepare for Mock Demo	Prepare for Mock Demo	Prepare for Mock Demo
4/19		Do Mock Demo, any final tweaks for board	Do Mock Demo, any final tweaks for server/API/front-end
4/26	Work on final design paperwork	Work on final design paperwork	Work on final design paperwork
5/3	Turn in final design paperwork and do final presentation	Turn in final design paperwork and do final presentation	Turn in final design paperwork and do final presentation

4 Ethics & Safety

The ethical or safety issues with our project pertain to the physical tray itself, and the microcontroller and Wi-Fi chips.

Citing the IEEE Code of Ethics #9 [6] - to avoid injuring others, their property, reputation, or employment by false or malicious action, we will work to ensure that the construction of our tray is structurally sound such that a user will not be concerned with electrical hazards such as exposed wires or static shock, or any harm from burning ICs or plastic. We will also make considerations to prevent damage to the tray's main circuitry by contact with user ingredients. (These considerations would take the form of a protective layer on the top of the board that prevents any spillage into the sensitive electronics underneath)

Citing the ACM Code of Ethics 2.9 [7], the greatest source of ethical and security concerns is the database itself. We will be allowing multiple users the ability to request data which will include information, specifically an email address. This could lead to bad actors stealing this data and targeting users [8]. To avoid these concerns, we will be hosting our server on DigitalOceans, which comes with its own security measures to prevent bad actors. Further, we will have our own authentication measures to do our best to prevent hacking.

An additional source of safety concern is the user-facing application, specifically in regards to the ACM Code of Ethics 2.9 [7]. While we expect the user of our prototype to load the application from source code provided by the designers, bad actors could potentially hijack API calls in the app itself to download malware onto a user's phone [9]. These concerns, while valid, are an extremely low risk as our application will not be downloaded outside of the authors knowledge for the duration of the project. Further, we will adopt a one-origin policy to authenticate the requests, since our HTTP protocols will only be handled by our one server.

References

[1] Avia Semiconductor Ltd. “24-Bit Analog-to-Digital Converter (ADC) for Weigh Scales” HX711 datasheet, 2012, v2.0. Available: https://cdn.sparkfun.com/assets/b/f/5/a/e/hx711F_EN.pdf [Accessed: 13-Feb-2020]

[2] HT Sensor Technology Co. Ltd. “TAL221 Miniature Load Cell” TAL221 datasheet, (n.d.) Available: https://cdn.sparkfun.com/assets/9/9/a/f/3/TAL221.pdf [Accessed: 13-Feb-2020]

[3] HT Sensor Technology Co. Ltd. “TAL220B Miniature Load Cell” TAL220B datasheet, (n.d.) Available: https://cdn.sparkfun.com/assets/e/5/f/5/6/TAL220B.pdf [Accessed: 13-Feb-2020]

[4] Espressif Systems “ESP8266EX Datasheet” ESP8266EX datasheet, 2019, v6.3. Available: https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf [Accessed: 13-Feb-2020]

[5] Microchip Technology “megaAVR® Data Sheet” ATMega328P datasheet, 2018 http://ww1.microchip.com/downloads/en/DeviceDoc/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061A.pdf [Accessed: 9-Feb-2020]

[6] ieee.org, "IEEE IEEE Code of Ethics", 2016. [Online]. Available: http://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 13-Feb-2020].

[7] Acm.org. (2020). *The Code affirms an obligation of computing professionals to use their skills for the benefit of society.*. [online] Available at: https://www.acm.org/code-of-ethics [Accessed 13 Feb. 2020].

[8] Digital Guardian. (2020). *The History of Data Breaches*. [online] Available at: https://digitalguardian.com/blog/history-data-breaches [Accessed 14 Feb. 2020].

[9] Chicago Tribune. (2020). *Chicago Tribune - Widespread Android Virus Could Hide in Popular App Updates*. [online] Available at: https://www.chicagotribune.com/news/ct-xpm-2012-04-16-sns-201204161154usnewsusnwr201204130413androidapr16-story.html [Accessed 13 Feb. 2020].

[10] Schanes, K., Dobernig, K. and Gözet, B. (2018). *Food waste matters - A systematic review of household food waste practices and their policy implications*. [online] Science Direct. Available at: https://www.sciencedirect.com/science/article/pii/S0959652618303366 [Accessed 12 Feb. 2020].

[11] Kopczynski, T. and Ness, D. (n.d.). *Five factors that can affect your weighing system's accuracy*. [online] Automation.com. Available at: https://www.automation.com/library/articles-white-papers/process-weighing/five-factors-that-can-affect-your-weighing-systems-accuracy [Accessed 13 Feb. 2020].

[12] Vijayaraghavan, G., Brown, M. and Barnes, M. (2008). *Electrical noise and mitigation – Part 1: Noise definition, categories and measurement*. [online] EE|Times. Available at: https://www.eetimes.com/electrical-noise-and-mitigation-part-1-noise-definition-categories-and-measurement/ [Accessed 13 Feb. 2020].

[13] Load sensor info - https://www.adminstrumentengineering.com.au/determining-load-cell-accuracy (Needs formatting)