

Theta Beta Engineer Project Report

Foundation Color Identifier and Dispenser
University of California, Irvine

Akhil Nandhakumar, Allyson Lay, Dalen Avrin Smith,
Elizabeth Yancey, Emma Shin, Harmeet Singh, Ival Momoh,
Jay Kim, Richard Tokiyeda, Victoria Sun

November 17, 2025

¹ Contents

² 1 Abstract	¹
³ 2 Introduction	²
⁴ 2.1 Research	²
⁵ 2.1.1 Background of Problem	²
⁶ 2.1.2 Existing Solutions	²
⁷ 2.2 Design Choices	³
⁸ 2.2.1 Past Considerations and Scrapped Plans	³
⁹ 3 Hardware Design and Specifications	⁴
¹⁰ 3.1 First Iteration	⁵
¹¹ 3.1.1 Initial CAD Model and Hand Sketches	⁵
¹² 3.1.2 Commentary	⁷
¹³ 3.2 Second Iteration	⁸
¹⁴ 3.2.1 Updated CAD Models	⁸
¹⁵ 3.2.2 Commentary	⁹
¹⁶ 3.3 Third Iteration	¹⁰
¹⁷ 3.3.1 Final CAD Models	¹⁰
¹⁸ 3.3.2 Final Manufacturing Drawings for Custom Parts	¹⁴
¹⁹ 3.3.3 Commentary	¹⁴
²⁰ 4 Software Design	¹⁵
²¹ 4.1 First Iteration	¹⁵
²² 4.1.1 Preliminary Diagrams	¹⁵
²³ 4.1.2 User Flow Diagram	¹⁸
²⁴ 4.1.3 Techstack	¹⁸
²⁵ 4.2 Second Iteration	¹⁹
²⁶ 4.2.1 Lo-fi/Mid-fi Designs	¹⁹
²⁷ 4.2.2 Basic Logic and Functionality	¹⁹
²⁸ 4.2.3 Core Features of Algorithm	¹⁹
²⁹ 4.3 Third Iteration	¹⁹
³⁰ 4.3.1 Final Product	¹⁹
³¹ 5 Electrical Systems Design	²⁰
³² 5.1 First Iteration	²¹
³³ 5.1.1 Initial Wiring Diagrams	²¹

34	5.1.2 Initial Power Calculations	22
35	5.2 Second Iteration	22
36	5.2.1 Circuit Building	22
37	5.3 Third Iteration	22
38	5.3.1 Final Wiring Diagrams	22
39	5.3.2 Final Power Calculations	23
40	5.3.3 Circuitry	23
41	6 Final Product	24
42	6.1 Testing Protocol	24
43	6.2 Integration	24
44	6.3 System Performance	24
45	7 Discussion	25
46	7.1 Limitations	25
47	7.2 Future Work	25
48	8 Conclusion	26
49	8.1 Bill of Materials	26
50	References	27

51 List of Figures

52	3.1 First Sketch: Housing	5
53	3.2 First CAD Models: Housing	6
54	3.3 First Sketch: Dispenser Systems	6
55	3.4 First CAD Models: Dispenser Systems	7
56	3.5 Second Iteration: Full Assembly	8
57	3.6 Second Iteration: Exploded View	8
58	3.7 Second Iteration: Housing Skeleton and Walls	9
59	3.8 Second Iteration: Brackets for Dispenser System	9
60	3.9 Final: Full Assembly <i>Exploded</i>	10
61	3.10 Housing: Walls	11
62	3.11 Housing: Lid	11
63	3.12 Housing: Skeleton	12
64	3.13 Dispenser System : Brackets - for connecting to housing, guide shaft, stabilizing syringe, and pushing down syringe plunger	13
65	3.14 Dispenser System : Full assembly	14
67	4.1 Initial Flowchart for Control Logic	15

68	4.2 Pseudocode: Image Processing Algorithms.	16
69	4.3 Pseudocode: Embedded System.	17
70	4.4 Initial Figma Mockup: UI/UX - Landing page.	17
71	4.5 Initial Figma Mockup: UI/UX - Loading the foundation shades.	18
72	4.6 User Flow Diagram	18
73	5.1 Preliminary Wiring Diagram	21
74	5.2 Final Wiring Diagrams	22

75 List of Tables

76	5.1 Power Calculations	22
77	8.1 Key Hardware Elements.	26

⁷⁸ **Chapter 1**

⁷⁹ **Abstract**

⁸⁰ The Foundation Identifier and Dispenser aims to develop a machine that is able to extract
⁸¹ samples from a picture of a human to determine the shade of their skin, allowing the machine
⁸² to dispense a corresponding foundation shade. The results produced should be both accurate
⁸³ and reproducible. Equipped with computer vision libraries and color correction algorithms,
⁸⁴ the system allows the user to take an picture alongside a reference color sheet, which has
⁸⁵ colors of known values. These captured values are processed to correct both camera bias,
⁸⁶ and lighting correction so that results may remain consistent regardless of lighting conditions
⁸⁷ during image capture. The system converts RGB values to LAB values, which are higher in
⁸⁸ accuracy in physical color mixing, as opposed to RGB, which is used to describe pixel colors.
⁸⁹ The program calculates how much of each color is needed to recreate the user's skin pigment.
⁹⁰ These pigments are then dispensed via a mechanical system comprised of a Raspberry Pi,
⁹¹ servo motors, and syringes. This project demonstrates an application of computer vision in
⁹² the cosmetic market to alleviate the burden of overconsumption and promote inclusivity.

⁹³ **Chapter 2**

⁹⁴ **Introduction**

⁹⁵ **2.1 Research**

⁹⁶ **2.1.1 Background of Problem**

⁹⁷ Testing foundation colors can be a frustrating experience for many, who are unable to
⁹⁸ find the perfect balance. At the end of the day, no line of foundation can realistically provide
⁹⁹ colors that cater to every possible skin tone. The seemingly unresolvable desire for a perfect
¹⁰⁰ shade leads many makeup users to spend hundreds on shades that are "close enough." This
¹⁰¹ leads to lots of waste, not just in money, but in bottles thrown out after purchase because
¹⁰² the match was ultimately unsatisfying.

¹⁰³ The beauty industry produces 120 billion packaging units per year and 95% of these units are discarded, as opposed to recycled [1]. In addition, traditionally a custom skin matched foundation can cost anywhere from \$60-100 per bottle, which leaves the consumers with the dilemma of whether they should take the gamble on the bottle that's just almost right versus breaking their wallet on a hand matched bottle. Our custom mixer machine offers the same accuracy in skin tone while also promoting cheaper makeup, sustainability, and waste-reduction.

¹¹⁰ Our foundation color picker abandons the concept of creating a set of discrete skin tones to choose from, instead, opting for custom mixed shades depending on the skin color detected by a picture. It also offers the unique ability to sample a shade without having to commit to a full sized bottle.

¹¹⁴ **2.1.2 Existing Solutions**

¹¹⁵ BoldHue provides an option for an AI powered foundation mixer. It matches skin tone through a smart wand that detects color. What they promised was millions of shades and the ability to store user's shades in profiles. However, their color detection methods provide room for lots of error because color isn't perceived the same depending on the lighting of the room, as well as the high cost of their machine.

¹²⁰ Our product will have built in lighting correction that removes biases from the camera and uses the Macbeth Color Reference sheet as a set of control colors. The reference sheet

₁₂₂ is what will allow the machine to recalibrate it's understanding of what colors are being
₁₂₃ percieve.

₁₂₄ 2.2 Design Choices

₁₂₅ 2.2.1 Past Considerations and Scrapped Plans

¹²⁶ Chapter 3

¹²⁷ Hardware Design and Specifications

¹²⁸ 3.1 First Iteration

¹²⁹ 3.1.1 Initial CAD Model and Hand Sketches

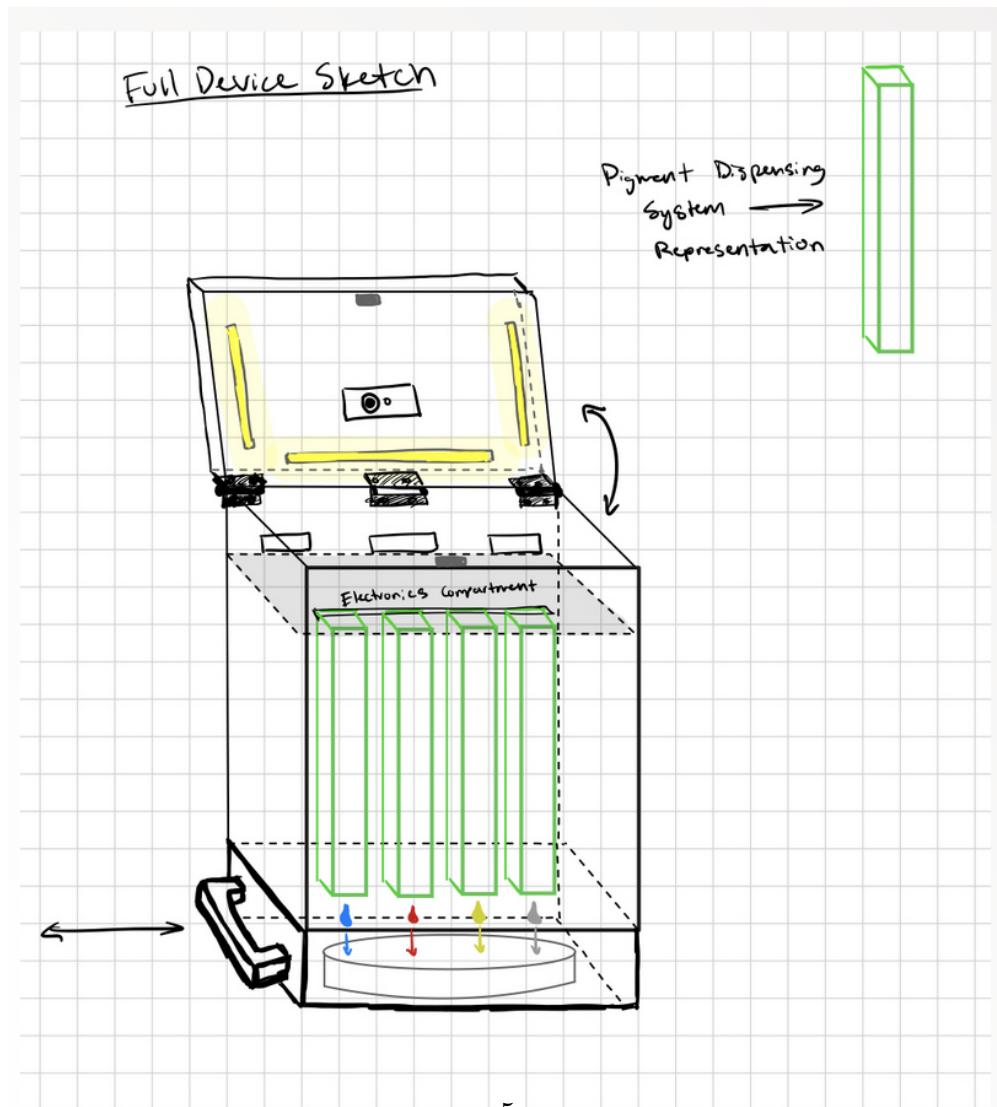


Figure 3.1: First Sketch: Housing

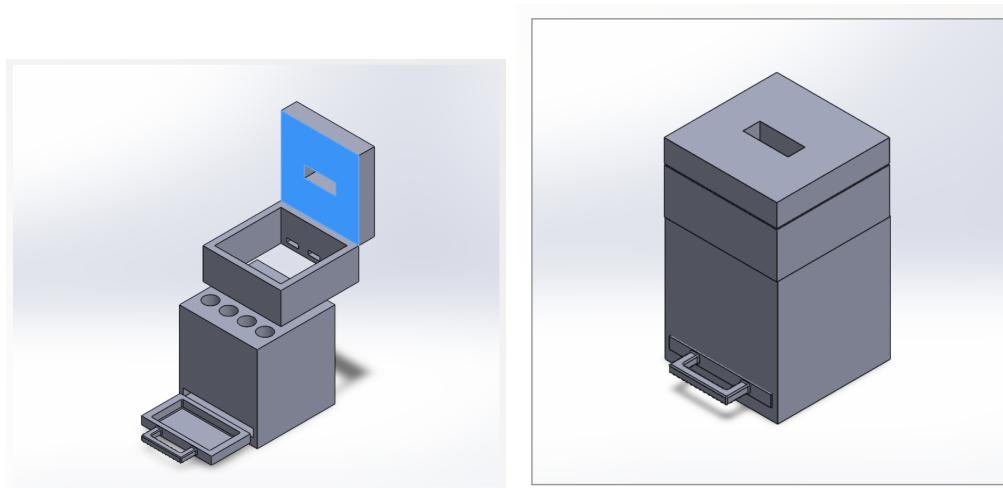


Figure 3.2: First CAD Models: Housing

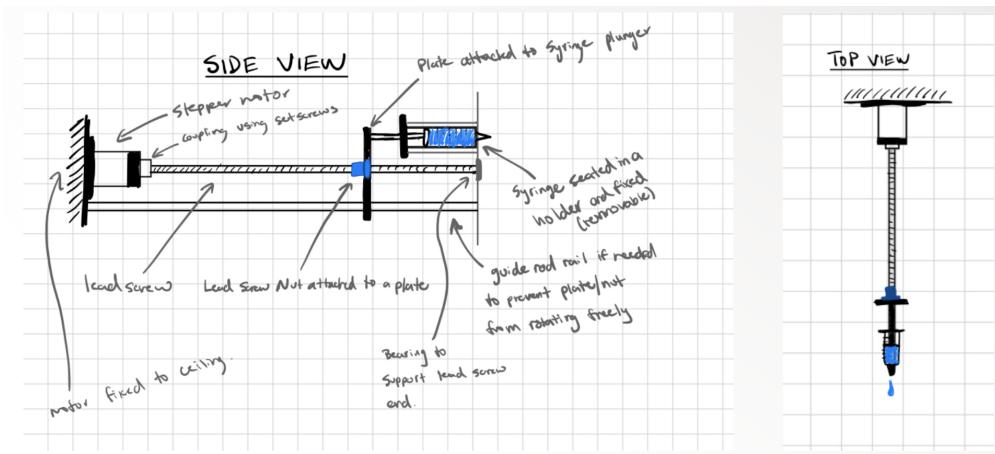


Figure 3.3: First Sketch: Dispenser Systems



Figure 3.4: First CAD Models: Dispenser Systems

¹³⁰ 3.1.2 Commentary

¹³¹ Our first CAD Models were very simple and provide the most general idea we had at
¹³² our projects conception. We made large changes to the design of the housing because the
¹³³ dimensions would have been much wider if we lined the syringes up, as shown in Figures 3.1
¹³⁴ and 3.2.

¹³⁵ The preliminary sketch we have of the dispenser system has stayed consistent throughout
¹³⁶ development.

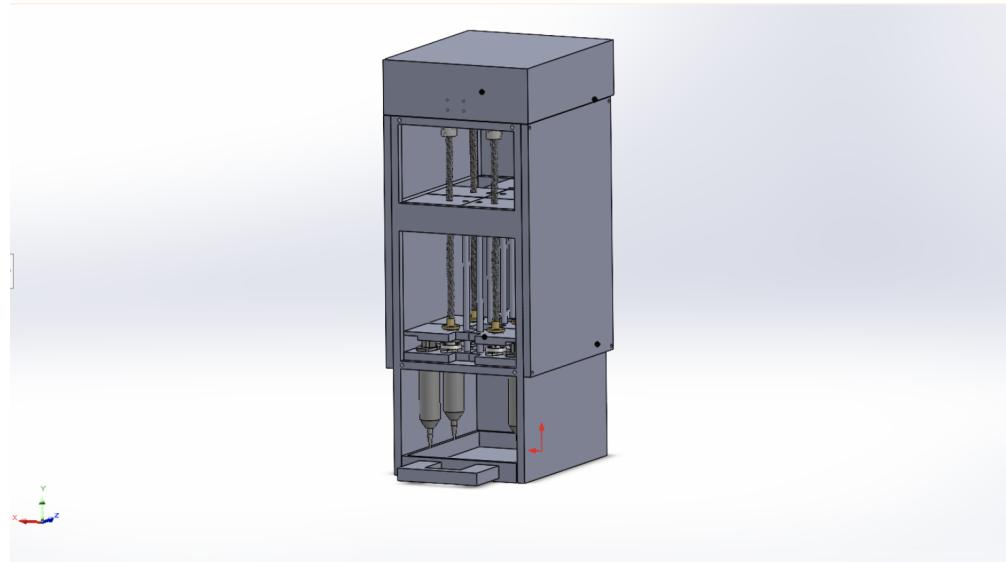
137 3.2 Second Iteration**138 3.2.1 Updated CAD Models****139 Assembly**

Figure 3.5: Second Iteration: Full Assembly

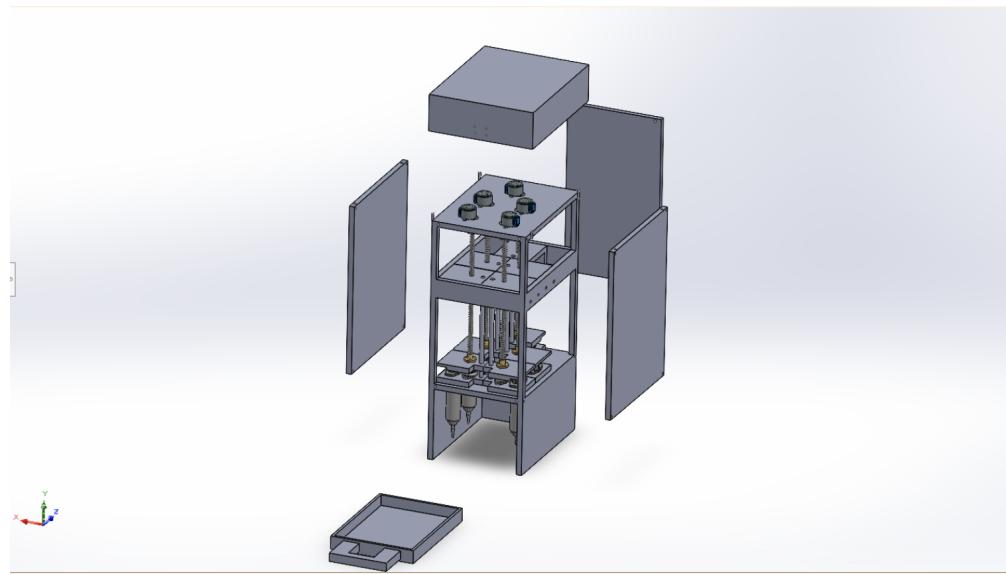


Figure 3.6: Second Iteration: Exploded View

140

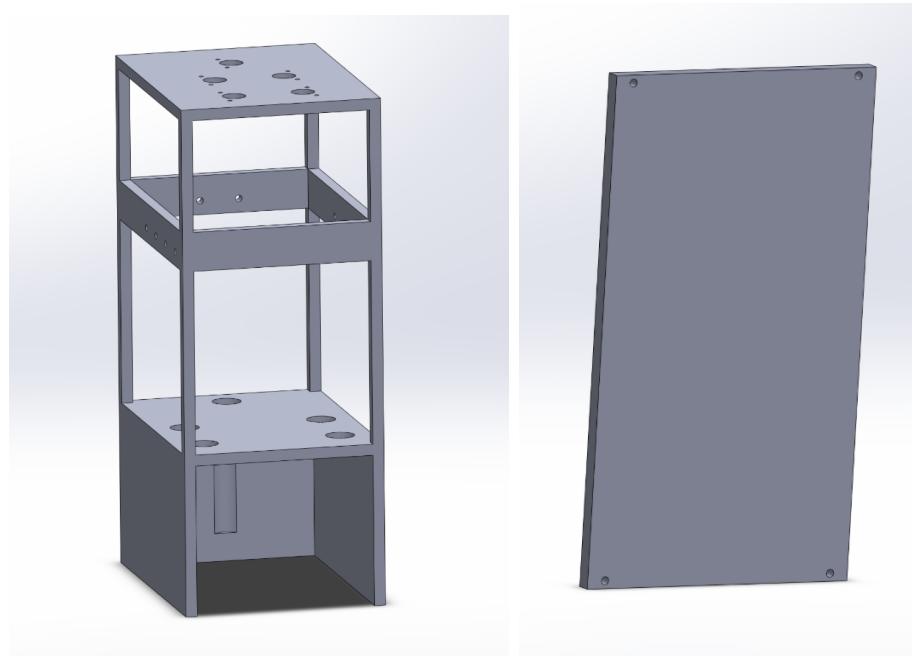
Housing

Figure 3.7: Second Iteration: Housing Skeleton and Walls

141

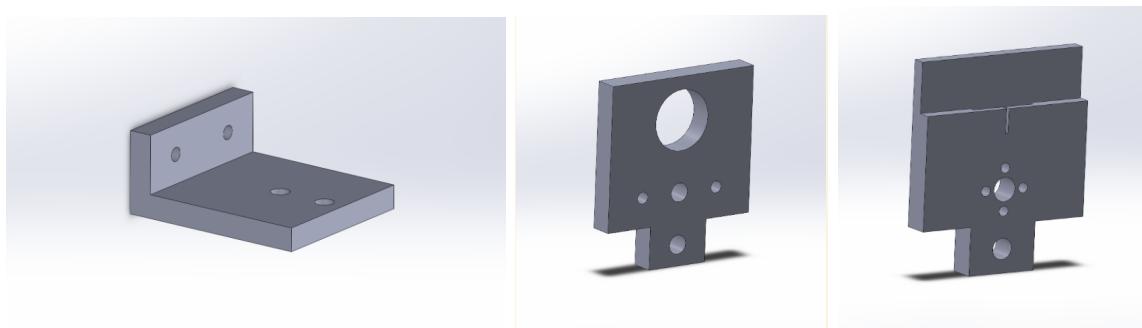
Dispenser

Figure 3.8: Second Iteration: Brackets for Dispenser System

3.2.2 Commentary

142

Housing

143

The housing skeleton is what holds the syringes in place while the walls create an enclosure so that the internal structure may be protected and hidden from users. The main concern with this iterations design was with the housing skeleton's stability. Since we were 3D printing all parts for the demo, the empy spaces could be flimsy, so the final iteration includes a redesign.

149 Dispenser

150 These brackets hold and control the dispenser system. The first L-shaped Bracket holds
151 the guide shaft in place. The other two are for stabilizing and pushing the syringe plunger
152 down. This second iteration is missing one of the parts but it will be shown in the next
153 section, along with final drawings of all parts.

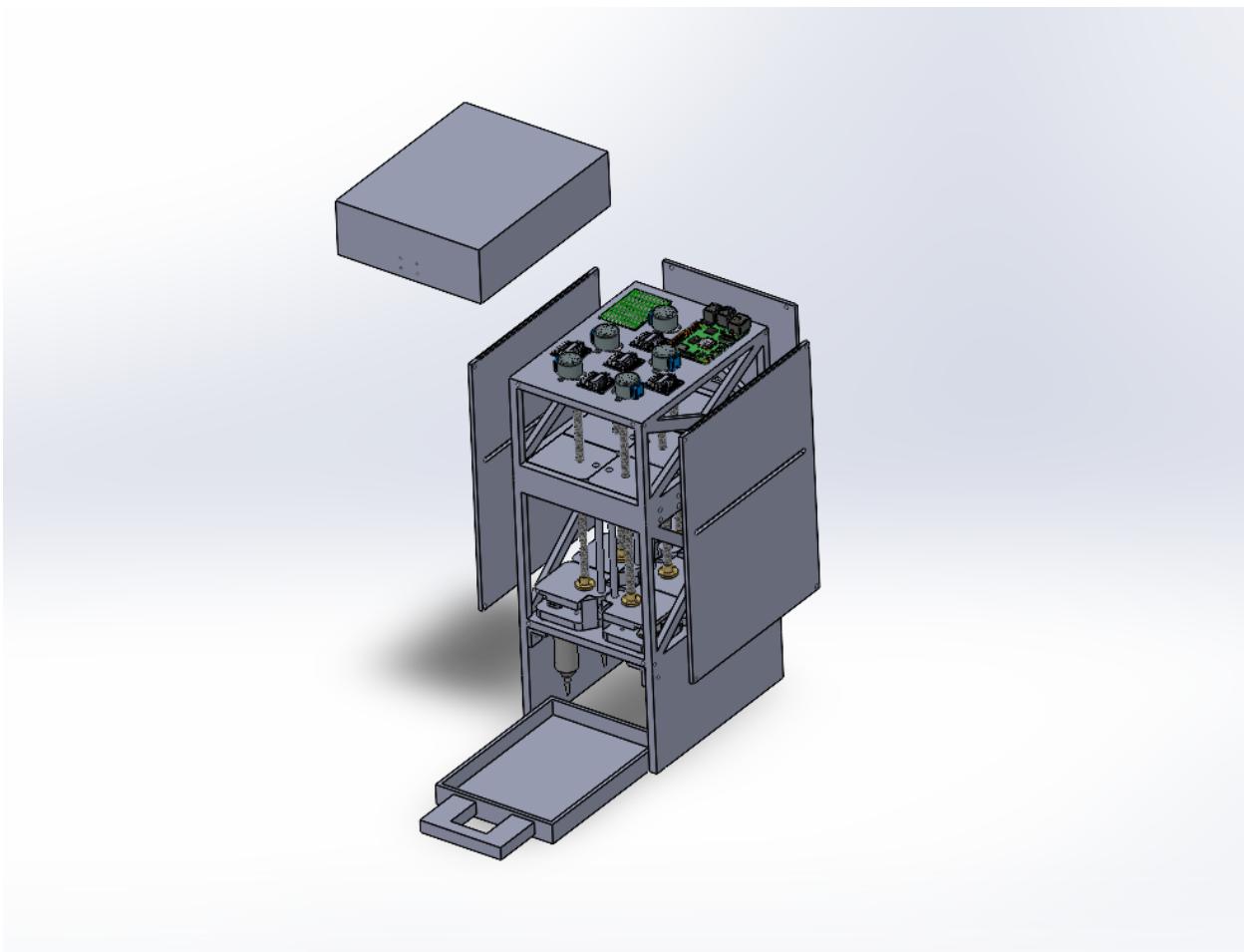
154 3.3 Third Iteration**155 3.3.1 Final CAD Models****156 Assembly**

Figure 3.9: Final: Full Assembly *Exploded*

157

Housing

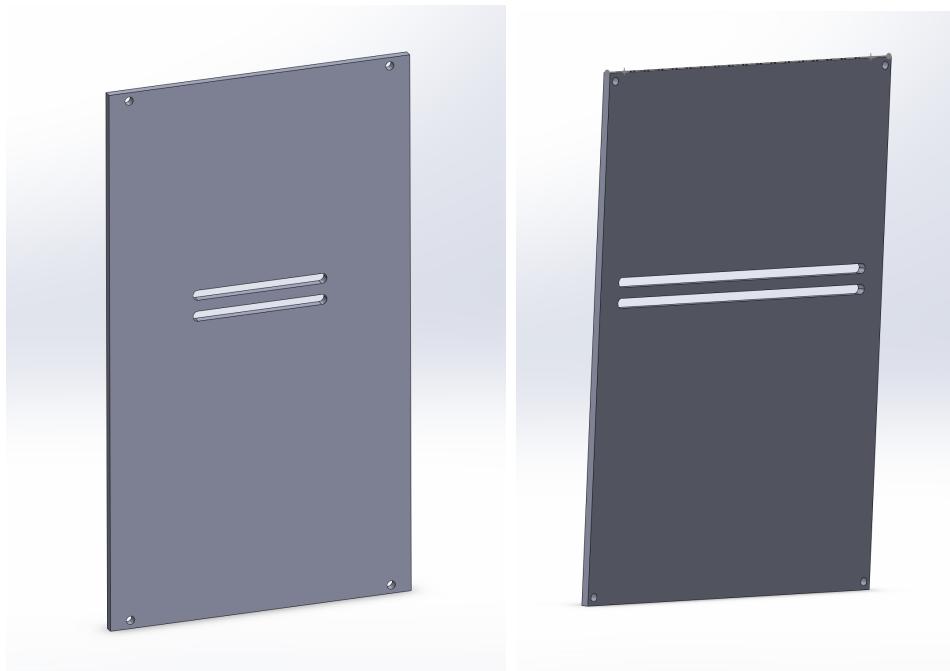


Figure 3.10: Housing: Walls

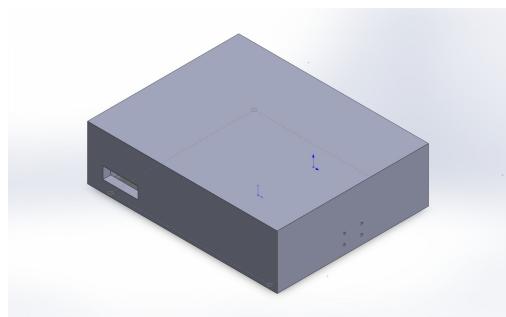


Figure 3.11: Housing: Lid

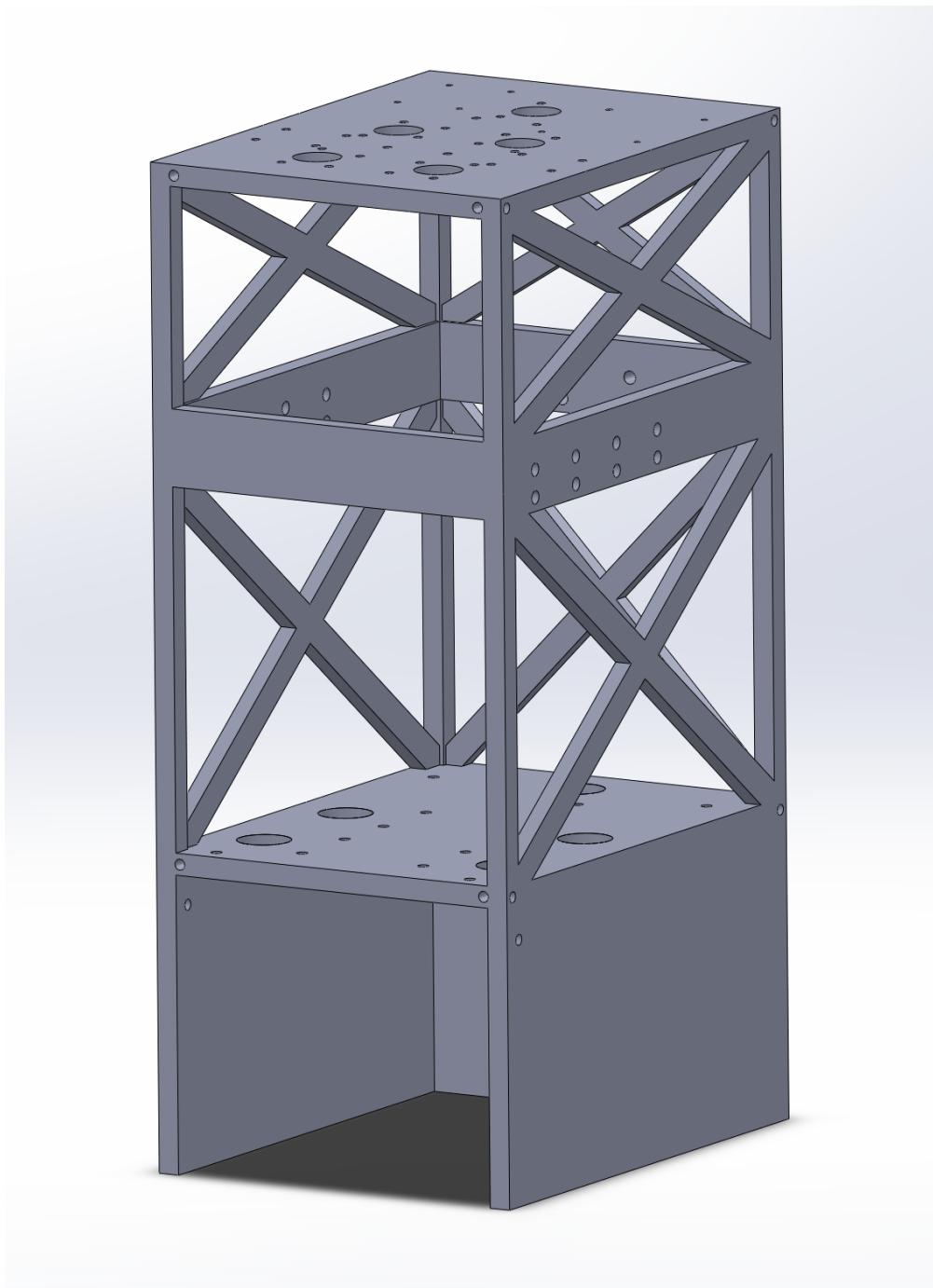


Figure 3.12: Housing: Skeleton

158

Dispenser

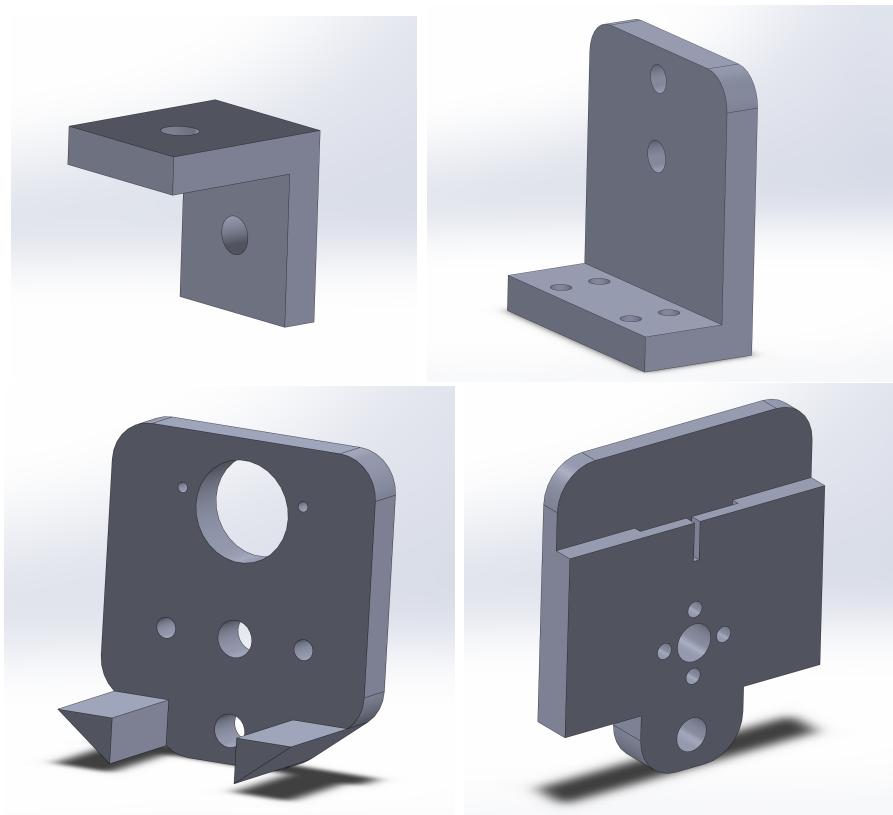


Figure 3.13: Dispenser System : Brackets - for connecting to housing, guide shaft, stabilizing syringe, and pushing down syringe plunger

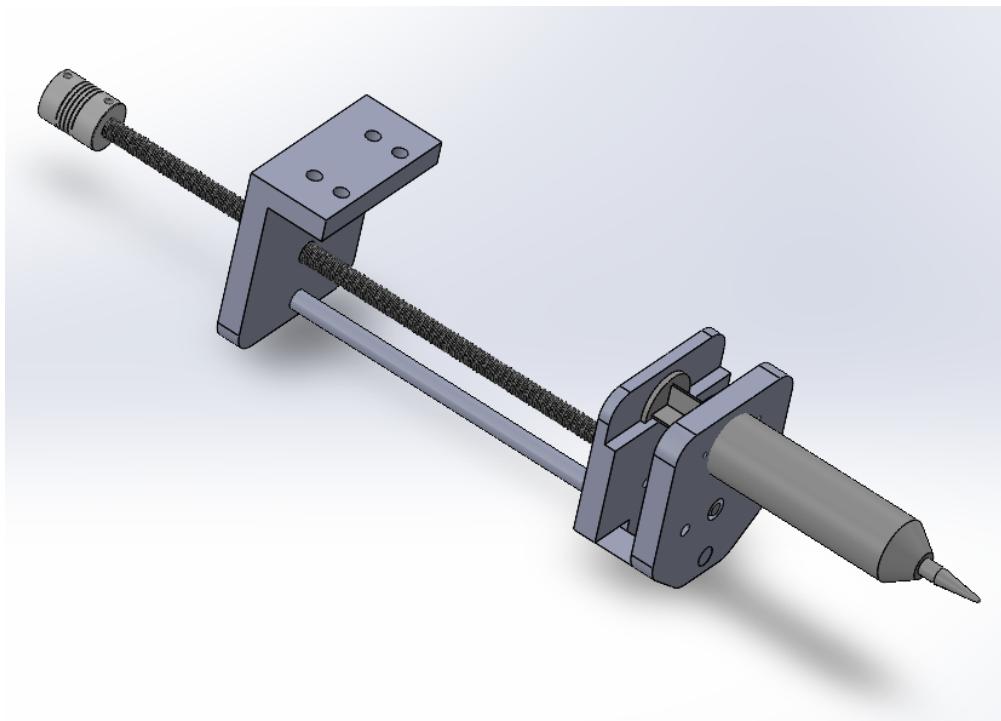


Figure 3.14: Dispenser System : Full assembly

¹⁵⁹ **3.3.2 Final Manufacturing Drawings for Custom Parts**

¹⁶⁰ **3.3.3 Commentary**

₁₆₁ Chapter 4

₁₆₂ Software Design

₁₆₃ 4.1 First Iteration

₁₆₄ 4.1.1 Preliminary Diagrams

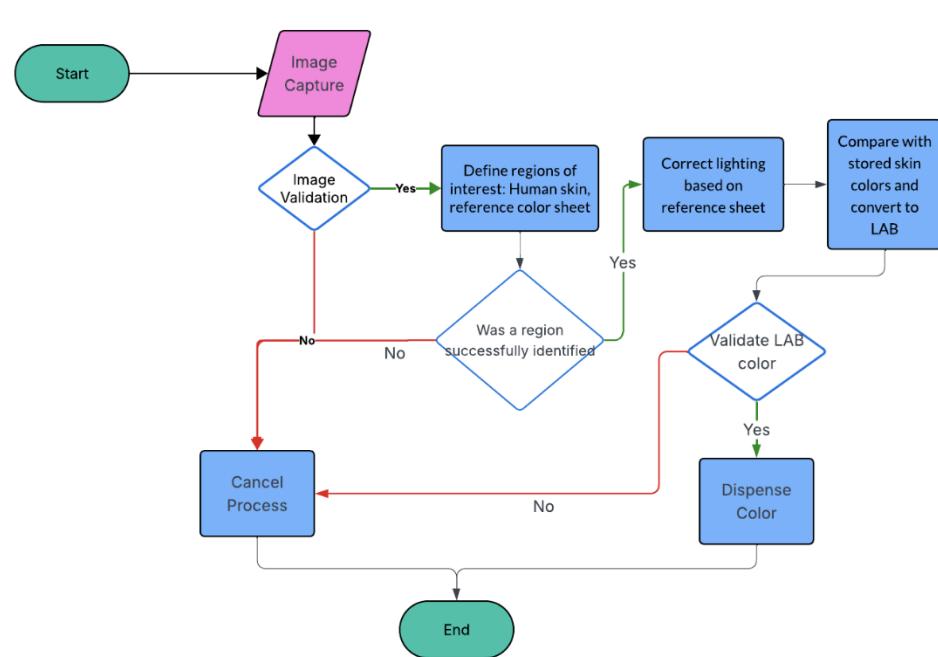


Figure 4.1: Initial Flowchart for Control Logic

```

# 1: Color Bias Adjustment and Sampling
```python
def IMAGE_PROCESSING(SAMPLE_PICTURE) :
 #INPUT: SAMPLE_PICTURE - image captured by camera
 #OUTPUT: LAB_values - color values compatible with paint mixing

 ### Get/Validate image containing skin region
 ### & reference colors/control sheet

 if SAMPLE_PICTURE is EMPTY/NOT_DETECTED :
 RAISE_ERROR "Image captured failed or canceled"
 return NULL

 # initial samples contain gamma-corrected RGB values
 # use OpenCV region of interest rectangle

 SKIN_SAMPLE = list of pixel RGB values that constitute a skin region from SAMPLE_PICTURE
 CONTROL_SAMPLE = list of pixel RGB values that encompass the reference color sheet SAMPLE_PICTURE

 # check whether the reference sheet is present or the image is too dark/light

 if CONTROL_SAMPLE is EMPTY/NOT_DETECTED :
 RAISE_ERROR "Control sheet not detected. Take picture with color reference sheet in a well-lit room."
 return NULL

 # get average gamma-corrected RGB codes for the skin region and control

 SKIN_RGB_AVG = calculate average of SKIN_SAMPLE
 CONTROL_MEASURED_VALUES = list of averages of CONTROL_SAMPLE

 # get linear RGB codes from gamma-corrected RGB codes
 # allows linear operations to be performed on the codes

 SKIN_LINEAR_RGB = apply reverse gamma-correction to SKIN_RGB_AVG
 CONTROL_LINEAR_RGB = list of linearized RGB codes from CONTROL_MEASURED_VALUES

 # find difference in the control input RGB codes and stored RGB codes

 CONTROL_KNOWN_VALUES = load("CONTROL_DATABASE.csv") and linearize the codes
 CONTROL_TRANSFORM = find transformation matrix that makes CONTROL_LINEAR_RGB = CONTROL_KNOWN_VALUES * CONTROL_TRANSFORM

 # apply difference to the values found in the skin region for lighting correction

 XYZ = apply CONTROL_TRANSFORM to SKIN_LINEAR_RGB

 # convert rgb value to lab for later processing

 LAB_VALUES = convert XYZ color space to LAB(XYZ)

 for element in LAB_VALUES :
 if element is OUT_OF_RANGE :
 RAISE_ERROR "color conversion error. take a new picture."
 return NULL

 return LAB_VALUES
```

```

Figure 4.2: Pseudocode: Image Processing Algorithms.

```
# 2: Raspberry Pi Code

```python
def EVENT_HANDLER():
 #extract lab values
 SAMPLE_PICTURE = CAMERA_CAPTURE initiated by BUTTON_PRESS
 LAB_VALUES = IMAGE_PROCESSING(SAMPLE_PICTURE)

 #calculate servo turns for desired recipe
 PAINT_QUANTITIES = assign paint quantities indicating how many
 | | | | | times the servo should turn with TARGET_SHADE

 DEPLOY_TO_RASPBERRY_PI(PAINT_QUANTITIES)
 #servo motors move syringes
```

```

Figure 4.3: Pseudocode: Embedded System.

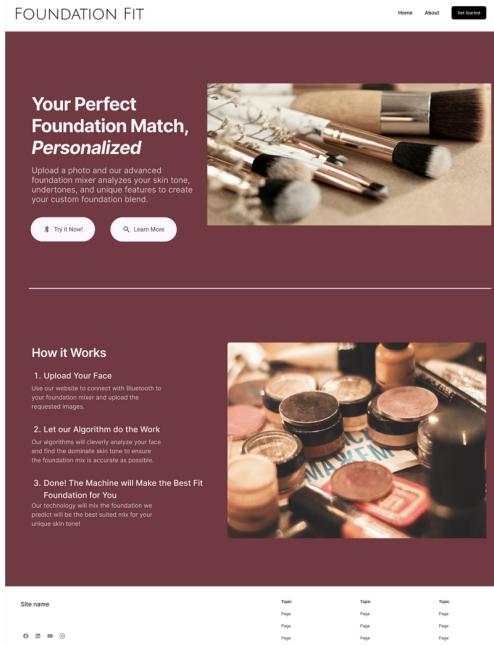


Figure 4.4: Initial Figma Mockup: UI/UX - Landing page.

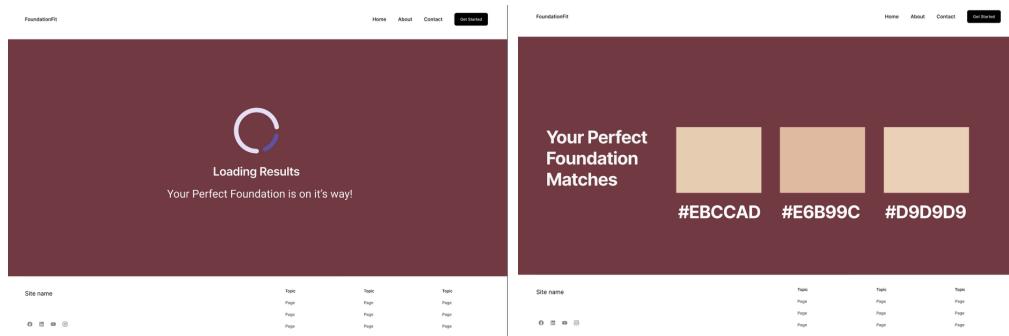


Figure 4.5: Initial Figma Mockup: UI/UX - Loading the foundation shades.

¹⁶⁵ 4.1.2 User Flow Diagram

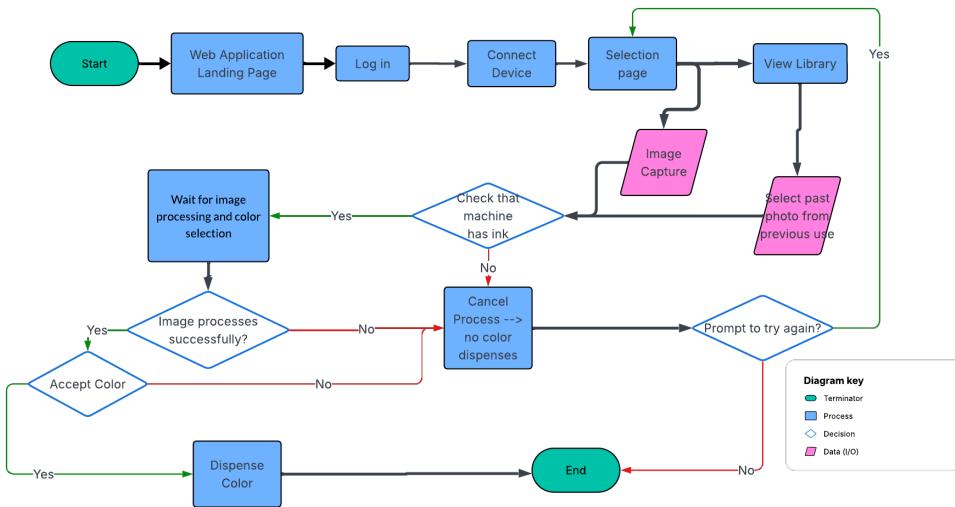


Figure 4.6: User Flow Diagram

¹⁶⁶ 4.1.3 Techstack

- ¹⁶⁷ Operating systems: Linux on Raspberry Pi
- ¹⁶⁸ Libraries: OpenCV, NumPy, Pandas
- ¹⁶⁹ Frameworks: React, Flask
- ¹⁷⁰ Languages: Python

¹⁷¹ **4.2 Second Iteration**

¹⁷² **4.2.1 Lo-fi/Mid-fi Designs**

¹⁷³ **4.2.2 Basic Logic and Functionality**

¹⁷⁴ **4.2.3 Core Features of Algorithm**

¹⁷⁵ **4.3 Third Iteration**

¹⁷⁶ **4.3.1 Final Product**

177

Chapter 5

178

Electrical Systems Design

179

5.1 First Iteration

180

5.1.1 Initial Wiring Diagrams

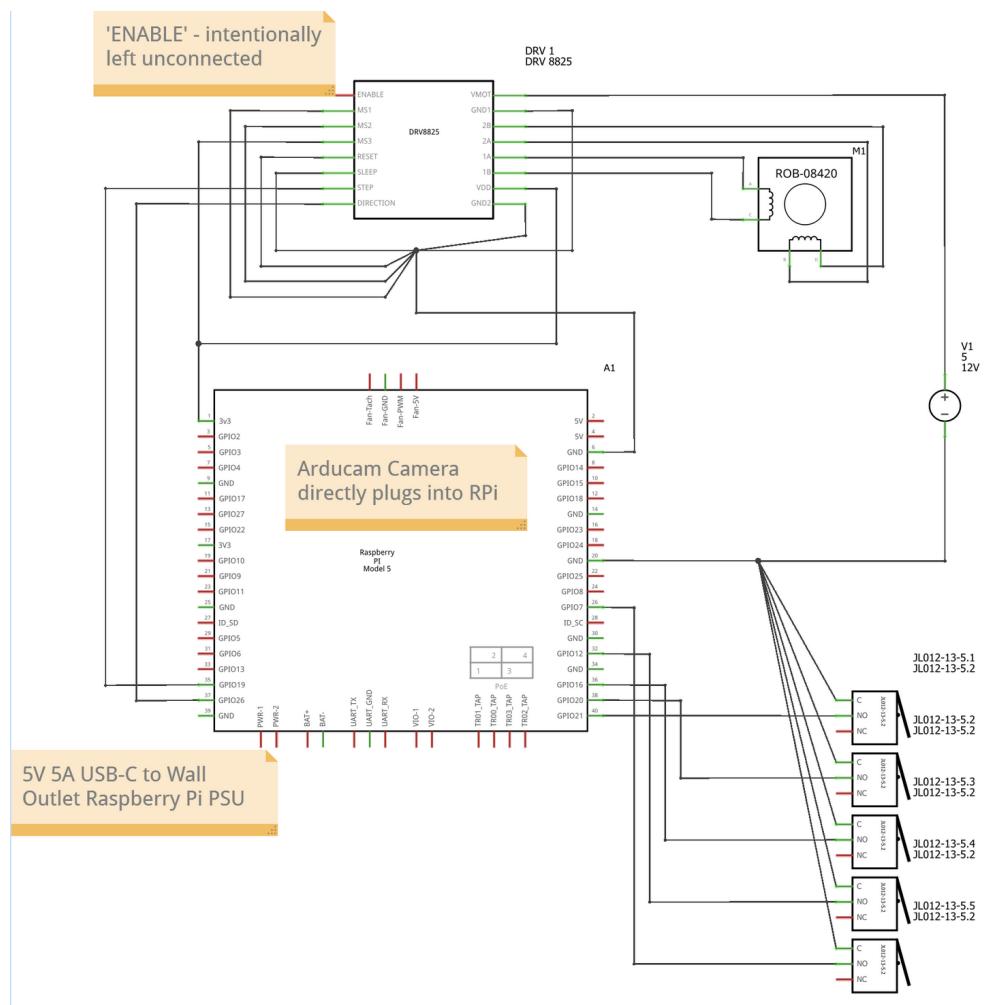


Figure 5.1: Preliminary Wiring Diagram

181 5.1.2 Initial Power Calculations

| Parameter | Symbol | Value |
|------------------|--------|-------|
| Phase Current | I | 0.7 |
| Phase Resistance | R | 4.0 |
| Phases | — | 2 |

Table 5.1: Power Calculations

$$P_{motor} = 2 * I^2 * R$$

$$P_{motor} = 2 * (0.7)^2 * 4.0 = 3.92 * 5 = 19.6W$$

$$P_{RaspberryPi} = 10W$$

$$P_{total} = 10 + 19.6 = 30W$$

185 5.2 Second Iteration

186 5.2.1 Circuit Building

187 5.3 Third Iteration

188 5.3.1 Final Wiring Diagrams

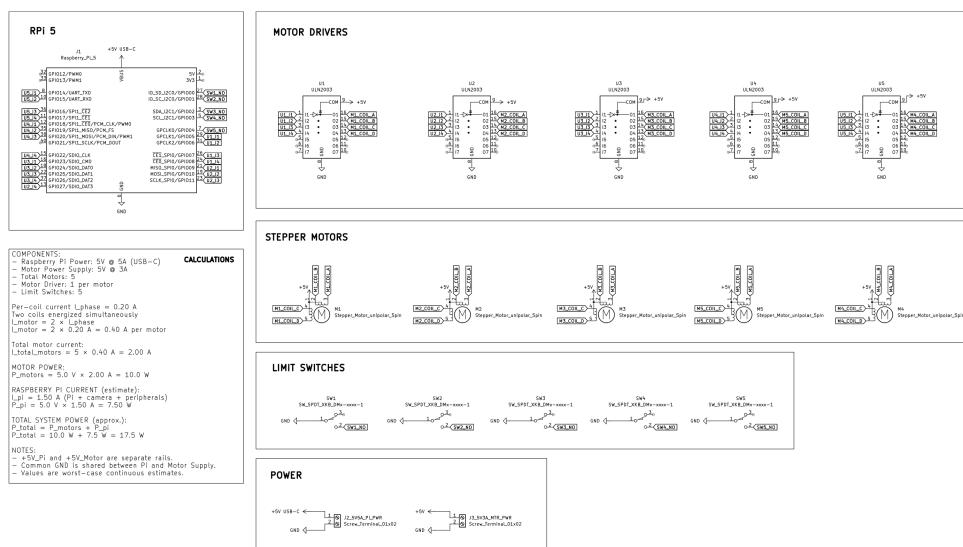


Figure 5.2: Final Wiring Diagrams

¹⁸⁹ **5.3.2 Final Power Calculations**

¹⁹⁰ **5.3.3 Circuitry**

₁₉₁ Chapter 6

₁₉₂ Final Product

₁₉₃ 6.1 Testing Protocol

₁₉₄ 6.2 Integration

₁₉₅ 6.3 System Performance

¹⁹⁶ Chapter 7

¹⁹⁷ Discussion

¹⁹⁸ 7.1 Limitations

¹⁹⁹ 7.2 Future Work

200 **Chapter 8**

201 **Conclusion**

202 **8.1 Bill of Materials**

| Component | Purpose | Quantity | Price |
|--|------------------------------|----------|-----------------|
| 2PC T8x8 Lead Screw with Brass Nut | Linear motion conversion | 3 | \$13.99 |
| 5-8mm Lead Screw Coupler 5 pack | Connect motor to screw | 3 | \$8.69 |
| 8x200mm Shaft Guide | Linear guidance | 3 | \$8.69 |
| 8mm Flange Mounted Pillow Block Bearing | Support rotating shaft | 3 | \$8.99 |
| Raspberry Pi 5 (4GB RAM) | Primary controller | 1 | \$66.00 |
| Plastic 30mL Syringes (Pack of 5) | Fluid handling prototype | 1 | \$6.99 |
| Raspberry Pi 5 Power Supply | Power for controller | 1 | \$15.99 |
| Assorted Metric Fasteners (M2–M5) | Mechanical assembly | 1 | \$20.00 |
| Limit Switch (10 pack) | Motion limit detection | 1 | \$5.99 |
| Arducam V3 Camera | Vision and monitoring | 1 | \$25.00 |
| Wiring Kit | Electrical connections | 1 | \$5.00 |
| Breadboard / Perfboard | Circuit prototyping | 1 | \$10.00 |
| 5V 3A DC Power Supply | General power source | 1 | \$7.47 |
| 5 Sets 28BYJ-48 Stepper + ULN2003 Driver | Secondary actuation system | 5 | \$14.99 |
| 1.75mm ABS Filament Reel (Black) | Structural printing material | 1 | \$16.99 |
| Total Estimated Cost | | | \$346.49 |

Table 8.1: Key Hardware Elements.

²⁰³ References

²⁰⁴ Bibliography

- ²⁰⁵ [1] Sophie Smith. Billions of beauty packaging goes unrecycled every year – theindustry.beauty. 2024. Accessed 2024.