

ECE 445

Spring 2025

Senior Design - Design Document

Skin Analyzer

Team 6

Ashley Herce, Waidat Bada, Shriya Surti

TA: John Li

Table of Contents

Introduction.....	3
1.1 Problem.....	3
1.2 Solution.....	3
1.3 Visual Aid(s).....	4
1.4 High-Level Requirements.....	4
Section 2: Design.....	5
2.1 Block Diagram.....	5
2.2 Electronics Subsystem.....	5
2.2.1 Power Subsystems.....	5
2.2.2 Sensor Subsystems.....	7
I. Color Sensor(s).....	7
2.3.3 Processing Subsystem Pulse Width Modulation.....	9
2.3 Software Subsystem.....	11
2.3.1 Color Analysis.....	12
Overview of Color Analysis.....	12
Undertone Analysis.....	14
2.3.2 Database Design.....	15
Overview of Database Design.....	15
Optimized Data Structure.....	16
2.4 Tolerance Analysis.....	17
2.4.1 XYZ to L*a*b* Conversion.....	17
2.4.2 Monk Skin Tone Scale Classification.....	18
2.4.3 Product Matching & Verification (Testing).....	19
Testing Procedure.....	20
2.4.4 Undertone Analysis.....	23
3.1 Cost.....	24
3.2 Schedule.....	25
Section 4: Ethics & Safety.....	28
4.1 Ethics.....	28
4.1.1 Algorithmic Fairness & Bias Mitigation.....	28
4.2 Safety.....	28
Section 5 References.....	29

Introduction

1.1 Problem

The beauty industry faces significant sustainability challenges, primarily stemming from systemic inefficiencies in accurately analyzing and matching skin tones to color products. These inefficiencies arise from several factors, including variations in lighting, differences in skin conditions, and subjective assessment methods. As a result, consumers often end up with mismatched products, leading to high return rates and increased environmental waste, as well as economic losses for brands.

In addition, the beauty industry has long favored lighter skin tones, and many product ranges are sparse or fail to adequately represent olive undertones and deeper shades. While shade ranges are gradually expanding, there is still a notable lack of inclusivity, especially for deeper skin tones. The commonly used Fitzpatrick scale, which classifies skin types based on sensitivity to UV exposure, underrepresents deeper shades, making it difficult for consumers with these skin tones to find products that truly match their complexion [1].

This lack of representation leads to increased frustration among consumers, particularly those with complex undertones or deeper complexions. Despite ongoing efforts to expand product ranges, the true distribution of makeup products contributes to a crisis of unmet needs and underrepresentation.

1.2 Solution

To address these challenges, we propose the development of an innovative colorimeter device that enhances beauty by accurately analyzing skin tone and recommending suitable products. Our device integrates a high-precision color sensor that measures the XYZ values of a person's skin across various lighting conditions. This method accounts for the variations in ambient lighting, melanin distribution, and undertones, providing a "true" and reliable reading of a person's skin tone, regardless of lighting inconsistencies.

By utilizing XYZ color space values, which are more universally consistent than traditional models like RGB or HSV, our device eliminates the inaccuracies often caused by lighting changes and skin condition variations. This allows for a more accurate match to makeup products, which is particularly important for individuals with deeper skin tones or undertones that are commonly overlooked by traditional beauty tools.

Our device is powered by an ESP32 microcontroller, which processes the data from the color sensor and transmits the results to a mobile app. Through the app, users can view personalized foundation matches and receive product recommendations from an extensive database that includes a diverse range of brands, shades, and price points.

Our device aims to challenge the industry's limited approach by mapping out a more comprehensive range of skin tones along the Monk skin tone scale, which emphasizes deeper and more inclusive shades. This will help reduce consumer frustration and raise awareness around the importance of better representation of skin tones in the beauty industry.

1.3 Visual Aid(s)

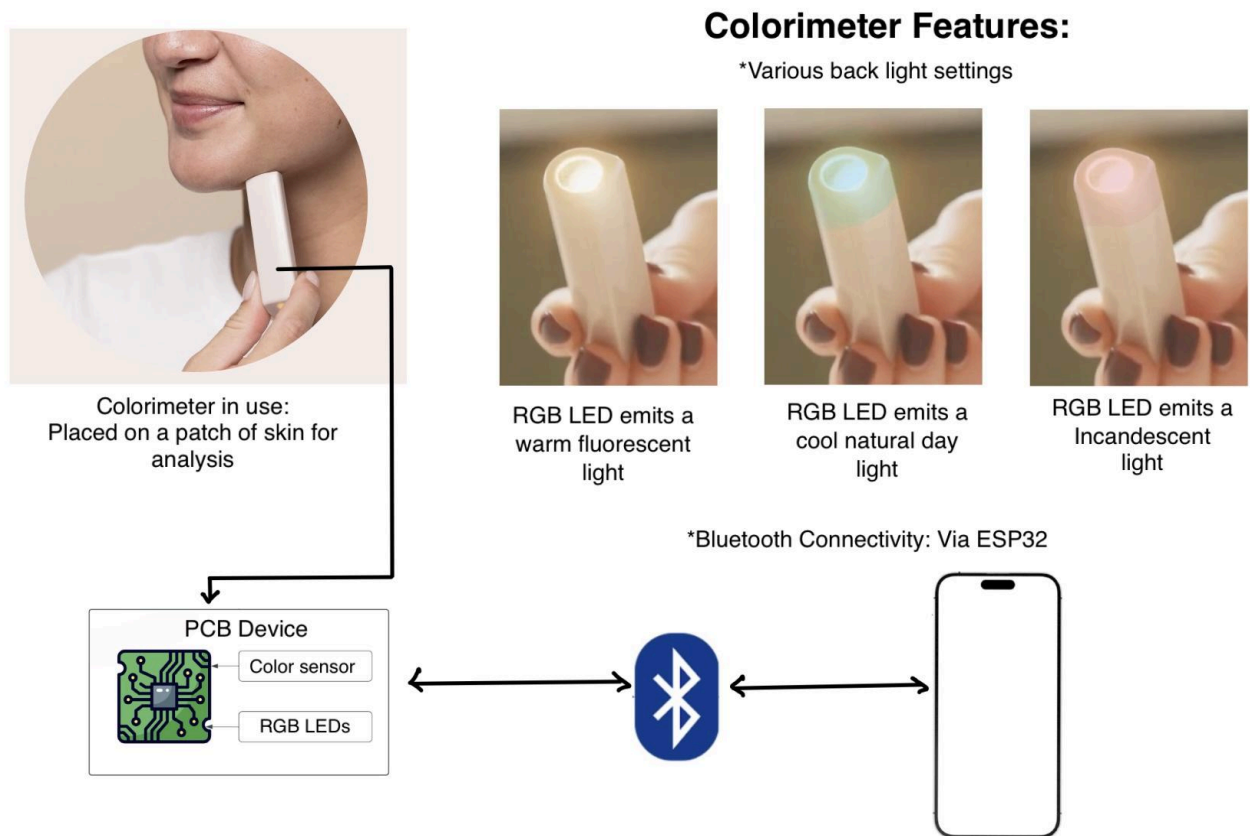


Figure 1.3.1: This is our theoretical visualization of our colorimeter device utilizing images from a similar functioning device called the BoldHue [3]

1.4 High-Level Requirements

- The system will replicate various ambient lighting conditions, including:
 - Warm Fluorescent – Represents indoor lighting.
 - Natural Daylight – Simulates daytime lighting.
 - Incandescent Light – Represents bulbs with warmer color temperatures.
- The Color Sensor will provide an accurate XYZ breakdown for any color with at least 85% accuracy using the CIELAB model and Monk scale.
- Using XYZ and $L^*a^*b^*$ values, the system will:
 - Classify the user's skin tone into one of the 10 Monk scale categories.
 - Determine the undertone (warm, neutral, or cool) by analyzing the red-to-blue ratio in the CIELAB color space.
 - Recommend foundation, skin tint, and/or concealer products within a $\Delta E \leq 1.75$ threshold from our database and display results via a mobile application.

Section 2: Design

2.1 Block Diagram

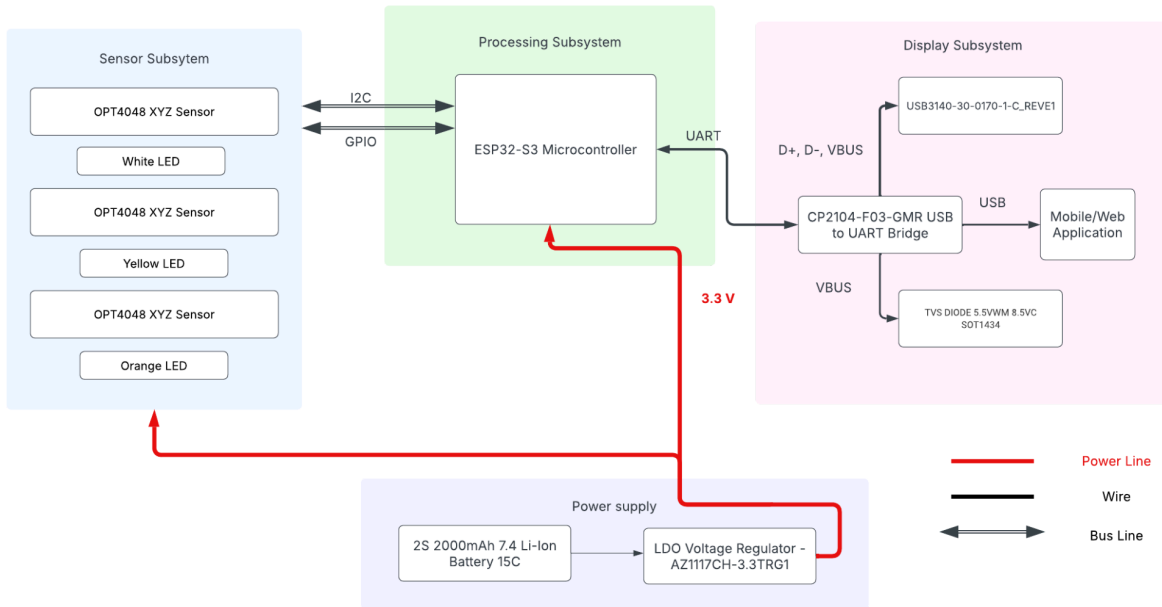


Figure 2.1.1: High-level breakdown of our colorimeter

2.2 Electronics Subsystem

2.2.1 Power Subsystems

Description & Purpose:

The power subsystem is designed to provide reliable and regulated power to the various components to ensure efficient operations. The color-matching device utilizes 3.3V and will provide power to the color sensor, RGB LEDs, and microcontroller.

Cell Choice Justification:

We will utilize a 2S 2000 mAh 7.4 Li-ion battery 15C (amazon) which consists of 2 lithium-ion cells in series with nominal voltages of 3.7V (7.4V total). The battery will provide 2 Amps of current for 1 hour before being fully discharged, assuming our device will be on for no longer than 10 minutes. For this battery, the C rating indicates the maximum safe continuous discharge rate. The maximum current can safely deliver 30 A [$2000\text{ mAh} * 15\text{C} = 30000\text{ mA}$].

Voltage Regulation Justification

We will utilize a low dropout converter to reduce battery voltage from 7.4 V to 3.3 V. We specifically chose an LDO over a buck converter as it offers better noise performance compared to a buck converter. Additionally, there is a small voltage difference between the input and output which will provide a faster load transient response to conserve energy when not in use.

To manage thermal output and ensure stable operation, a heat sink will be attached to the LDO voltage regulator. The LDO can generate significant heat, especially under high load conditions or larger voltage drops. This can lead to the overheating of the regulator and performance degradation or failure. The heat sink will dissipate the heat and keep the regulator within its safe operating temperature range.

Interactions:

The power subsystem interfaces with all electrical components, supplying regulated power (3.3v) to the microcontroller, sensors, and LEDs. By ensuring stable voltage and current delivery, it supports uninterrupted operation across the entire system.

Current Parts:

- 2S 2000 mAh 7.4 Li-ion battery 15C
- LDO Voltage Regulator AZ1117CH - 3.3TRG1

Requirements	Verification
Provide 30 A average/continuous current	<p>Use a low-resistance shunt resistor to measure voltage drop:</p> <ul style="list-style-type: none"> ● Connect the battery's positive terminal to the load through the shunt resistor ● Use a multimeter to measure the voltage drop (V) across the resistor ● Calculate current using Ohm's Law
Provide stable 3.3V \pm 5%	<ul style="list-style-type: none"> ● Use an oscilloscope to measure the output voltage ripple under various load conditions. ● We will ensure the ripple voltage stays within the \pm 5% tolerance band of the output through this methodology <ul style="list-style-type: none"> ○ Connect an oscilloscope probe to the 3.3V rail ○ Set the oscilloscope to AC coupling and measure the voltage ripple ○ Verify that ripple voltage does not exceed 50mV peak-to-peak
Shutdown safely when not actively using the device	<ul style="list-style-type: none"> ● Connect the power subsystem (battery, LDO regulator, and microcontroller) to the system under test ● Simulate active operation by running the microcontroller, LEDs, and sensor for a set period ● Issue a shutdown command via software or detect inactivity based on a predetermined timeout condition. ● Monitor the power output using a multimeter or oscilloscope to verify that voltage drops to 0V or enters a low-power state. ● Measure the current draw using a clamp meter or current sensor to confirm minimal power consumption (<X mA, based on design requirements). ● Ensure proper restart functionality by reactivating the device and verifying normal operation.
Maintain LDO regulator temperature below the maximum rating	<ul style="list-style-type: none"> ● Use a heat sink by attaching an appropriately sized heat sink to the LDO regulator to dissipate excess heat ● Apply thermal paste or thermal pad to provide heat transfer between the LDO and the heat sink ● Ensure good ventilation by placing the LDO in an area with airflow, avoiding enclosed spaces where heat can build up.

2.2.2 Sensor Subsystems

I. Color Sensor(s)

Description & Purpose:

The sensor subsystem is at the heart of our solution and is critical in detecting and determining one's skin tone. We need to be able to accurately quantify someone's skin color, including undertones, hues, etc, and use the data to properly provide them with their closest makeup shades across numerous brands. In order to do this we originally were looking at an RGB sensor, however, based on later justification we were able to find an XYZ sensor that would eliminate added computing time on chip. We have decided to use the OPT4048DTSR XYZ Color Sensor in order to do so. This color sensor requires an input of 3.3V which conforms with other aspects of our design as well including our processing subsystem. This sensor also allows us to establish I2C communication with the microcontroller in order to relay the acquired data to our users.

II. LED Sensor(s)

In order to find the true skin tone we also have LED sensors to provide different ambient lightings to mimic different environmental conditions skin could be under while being photographed/analyzed. These conditions include natural daylight, fluorescent lighting, incandescent lighting, etc. This will also take in 3.3V from the microcontroller and is provided information through software and GPIO pins on the ESP32.

Justification:

There are two types of color sensors, RGB and XYZ, where XYZ senses a complete standardized color space while RGB reads an unstandardized subset of the standard color space [4]. RGB color space is not perceptually uniform while XYZ is primarily based on human color perception and encompasses all perceivable colors. The Y component of the XYZ sensor corresponds to luminance allowing for more accurate brightness measurements and is more sensitive to color variations in comparison to the RGB.

We aim to measure skin tone in various lighting conditions, an RGB sensor often requires calibrations to convert the RGB coordinates to the correlated color temperature spectrum (Ambient Lighting). The XYZ sensor maps to the nearest CCT value without calibration making it an ideal color sensor for this subsystem. Additionally, converting from the XYZ color space to CIELAB is computationally less extensive compared to RGB to CIELAB (detailed in tolerance analysis).

Interactions:

This sub-system interacts with the processing and power subsystems of the circuit as the OPT4048 needs 3.3V in order for its own component to work and it is also connected to numerous GPIO pins on the microcontrollers to communicate using I2C. We use 3 GPIO pins per color sensor on the ESP32 microcontroller for SCL (clock), SDA (data), and INT (interrupt for I/O). We also require GPIO pins to program the LEDs as well.

Current Parts:

- OPT4048 XYZ Color Sensor
- RGB LED
- White LED
- Yellow LED
- Orange LED

Requirements	Verification
The OPT4048 color sensor must detect XYZ values with an accuracy of 85% compared to any/all standardized input colors.	<ul style="list-style-type: none"> ● Position the color sensor exactly 1 inch away from the color samples. ● Utilize XYZ values from predefined paint swatches and foundation shades as the testing reference. ● Capture and record the XYZ values detected by the sensor. ● Compare the detected values with the reference values by analyzing the individual X, Y, and Z components and calculating the differences between the experimental and actual values for each to determine if the accuracy meets or exceeds 85%.
The color detection latency must be less than 150 ms after the initial set up of I2C communication through the ESP32 microcontroller	<ul style="list-style-type: none"> ● Connect the XYX sensor to the microcontroller ● Connect the ESP32 to the computer connected to Arduino ● Set up a timer ● Start time right before I2C initialization begins ● After the first color detection takes place stop the timer and takes the difference to find the total latency of an instruction call to detect color
The LED must be able to switch ambient lighting conditions with a latency of less than or equal to 200 ms to mimic different environmental conditions	<ul style="list-style-type: none"> ● Set up the ESP32 microcontroller to control the LED system via GPIO pins. ● Connect an oscilloscope or logic analyzer to the LED power lines to measure the response time. ● Send a command from the ESP32 to switch the LED lighting mode from one condition to another. ● Record the time difference between the command initiation and the moment the LED reaches full intensity for the new lighting condition. ● Repeat the test for multiple transitions (e.g., daylight → fluorescent, fluorescent → incandescent). ● Verify that the latency for each transition is ≤ 200 ms.

2.3.3 Processing Subsystem Pulse Width Modulation

Description & Purpose:

Our processing subsystem is powered by the ESP32 microcontroller, which serves as the central intelligence of the system, managing communication between key components. It interfaces with the color sensors using the I2C protocol, ensuring seamless data exchange through shared SDA and SCL lines. Additionally, it controls the LED sensors via dedicated GPIO pins, utilizing the ESP32 LEDC (LED Controller) library to independently regulate each LED color through pulse width modulation (PWM) channels. Beyond sensor management, the ESP32 plays a critical role in system-wide coordination, connecting with both the power subsystem and the display/software subsystem. It dynamically adjusts RGB LED backlighting, processes raw XYZ data from the color sensors, and converts it into precise six digit HEX values to accurately classify skin tones and recommend suitable foundation products.

Furthermore, the ESP32 will transmit processed data to the display interface, enabling real-time visualization and analysis. With its capability to seamlessly integrate subsystems via both I2C and GPIO communication, coupled with low power consumption and wireless connectivity, the ESP32 stands as a highly efficient and dependable processing unit for our device.

Interactions:

The microcontroller will be powered by the power subsystem and used to send data from the sensor subsystems to the display subsystem. It will interact with the sensor subsystems via I2C and GPIO communication whilst interacting with the display subsystem via Bluetooth and UART-USB.

Requirements:

- Provide a stable and responsive Bluetooth connection

Current Parts:

- Bluetooth ESP32-S3 (MCU)

Requirements	Verification
The ESP32 must support high-speed I2C communication (up to 1 MHz) to efficiently acquire sensor data	Use an Oscilloscope or Logic Analyze: <ul style="list-style-type: none"> • Connect an oscilloscope or logic analyzer to the SCL line • Trigger a sensor read operation and measure the clock signal • Ensure the clock frequency matches the configured speed
The ESP32 must use LEDC PWM channels to independently control RGB LED brightness and color settings	Monitor LED brightness changes with a multimeter or oscilloscope to confirm proper PWM signal output
The ESP32 must process raw RGB data from the sensors and convert it into six-digit HEX values to determine skin tone	Compare the processed HEX values with a reference color chart to validate accuracy [2]
The ESP32 must establish a stable and responsive Bluetooth connection for real-time data transmission	Measure latency and packet loss using debugging tools such as ESP-IDF Bluetooth logs

2.3 Software Subsystem

Description & Purpose:

Our display subsystem will primarily be on an iOS application programmed with Swift. We will initialize the BLE service on the ESP32-S3 microcontroller to connect to an iOS device. Using either ESP-IDF (Espressif IoT Development Framework) or Arduino Framework we can make the ESP32-S3 discoverable by iOS Device and create a custom BLE Service to read, write, and connect to our device.

On the iOS development side, we will utilize CoreBluetooth Framework for BLE and scan for BLE peripherals to identify our device.

Using the kaggle cosmetic foundation shade cvs file we can create a viewable iOS application showcasing the Brand, product, Image, Shade name, and hex value. We will use Apple's Core Data framework to host the information and parse through the database using SQLite to fetch data.

Based on the Data provided by the color sensor, we will display to the user the following:

- List of products matching their hex value (if they exist)
- Undertones of user (Cool, Warm, Neutral, Olive, Undetermined)
- Classification of skin tone on Monk and Fitzpatrick Scales

Since our code is currently written in Python for testing, we aim to use PythonKit for the backend aspect of our IOS application to avoid unnecessary rewriting in Swift.

Justification:

Swift provides a robust and efficient foundation for the iOS application. Core Data with an SQLite database offers a scalable solution for managing data. Developing a mobile application extends the project beyond the course and project as well. We can further develop the mobile application scale the database after the course and allow Apple users to utilize the app with more applicable features we currently have not considered.

Interactions:

The display subsystem will interact with the processing subsystem (microcontroller) and display the data provided from the sensors to the screen. It will not directly interact with the sensor subsystem but will be responsible for showing the output of the sensors based on data given by the microcontroller. It will interact with the microcontroller via BLE and UART-USB

Current Parts:

- iOS Smart Device (v. 15 or higher)
- USB to UART Bridge
- SQLite3 Database & Kaggle Resource

Requirements	Verification
Display real-time CIELAB values on the screen	<ul style="list-style-type: none"> • Connect to ESP32 via BLE and verify communication using a terminal (e.g. PuTTY or Arduino Serial Monitor) • Confirm CIELAB values are updated by logging data from the color sensor and displaying it on the screen • Use a stopwatch or Python counter to ensure updates occur within 500 ms - 800 ms of receiving sensor input
Implement tolerance calculations for perceptually accurate product matching	<ul style="list-style-type: none"> • Use a dataset of Lab values to calculate ΔE^* values between predefined samples (target) and standards (shade values) • Verify the ΔE^* values are calculated accurately by comparing results to manual calculations via MATLAB • Confirm ΔE^* meets threshold of ≤ 1.75 for acceptable matches
Display product information that matches CIELAB values from the color sensor	<ul style="list-style-type: none"> • Set up PythonKit on Swift UI and verify data retrieval by printing fetched product information from the database • Perform a query using sensor output and confirm correct product information is displayed • Test communication between ESP32 and the database via BLE by sending queries and verifying responses using MicroPython scripts
Classify the user's skin tone into Monk and Fitzpatrick Scale	<ul style="list-style-type: none"> • Verify classification accuracy by testing shades labeled as 'fair', 'olive', or 'deep' against corresponding Monk and Fitzpatrick categories • Use KNN clustering and confirm that shades are categorized based on their closest Lab value(s) in the database • Display colors within $\Delta E^* \leq 1.75$ for visual confirmation
Ensure color sensors and database are fair and unbiased for multiple skin tones	<ul style="list-style-type: none"> • Create a stratified Testing Data that includes a representative skin tone samples across all 10 Monk Skin Tone categories • Measure the product matching accuracy separately for each Monk Skin Tone Accuracy • Compare the accuracy metrics across different skin tone categories. Significant differences in accuracy will indicate potential bias • Iteratively refine our algorithm if bias is detected by adjusting matching thresholds or incorporating additional features.

2.3.1 Color Analysis

Overview of Color Analysis

Figure (2.3.1) demonstrates the 10 categories of the Monk Skin Tone Scale along with the corresponding 'characteristics' from the Fitzpatrick Scale. The Fitzpatrick scale is an industry-standard used in

dermatology [5]. However, studies have highlighted that the Fitzpatrick scale underrepresents darker skin complexions [5]. The primary application of the Monk scale is in evaluating datasets for computer vision models.

While we will use the Monk scale for classification, the Fitzpatrick scale remains the ‘golden standard’ widely used in skin pigmentation research. For our purposes, we aim to classify users using both the Monk and Fitzpatrick scales: the Fitzpatrick scale is broadly adopted, while the Monk scale better represents a wider range of darker skin tones.

The "Total Count" column indicates the number of unique shades in our database that align with each specific category of the Monk scale. While we developed our database with a wide range of brands, there is still an underrepresentation of darker skin tones using the Monk Scale. This is a common issue within the beauty industry that we hope to address through our product.

Fitzpatrick	Monk Scale	Characteristics	Total Count
I	1	Very Fair: Always burns, never tans (palest; freckles); very light or white, "Celtic" type	76
	2		334
II	3	Fair: Usually burns, tans minimally (light colored but darker than pale); light or light-skinned European	105
	4		726
III	5	Medium: Sometimes mild burn, tans uniformly (golden honey or olive) light intermediate, or dark-skinned European	3881
	6		2626

IV	7	Olive: Burns minimally, always tans well (moderate brown); dark intermediate or "olive skin"	1291
	8		566
V	9	Brown: Very rarely burns, tans very easily (dark brown); dark or "brown" type	62
	10		16
VI		Dark Brown: Never burns (deeply pigmented dark brown to darkest brown); very dark or "black" type	
Total Shades:			9683

Figure (2.3.1) [Characteristics Reference [6]]

Undertone Analysis

Undertone is the color underneath the surface of the skin that reflects your skin tone's overall hue

To determine skin undertones using CIELAB color model, we focus on the a^* (red-green) and b^* (yellow-blue) chromaticity axes. Skin undertones are primarily identified through the a^* and b^* values, which reflect the balance of redness, yellowness, or neutrality.

Interpreting undertones: [6]

- **Warm Undertones (Red prominence):** positive a^* and lower b^* value indicates a higher red value and lower yellow undertone
- **Cool undertones (Yellow prominence):** positive b^* values and neutral to moderate a^* values
- **Neutral undertones:** Balanced a^* and b^* values near the central achromatic zone

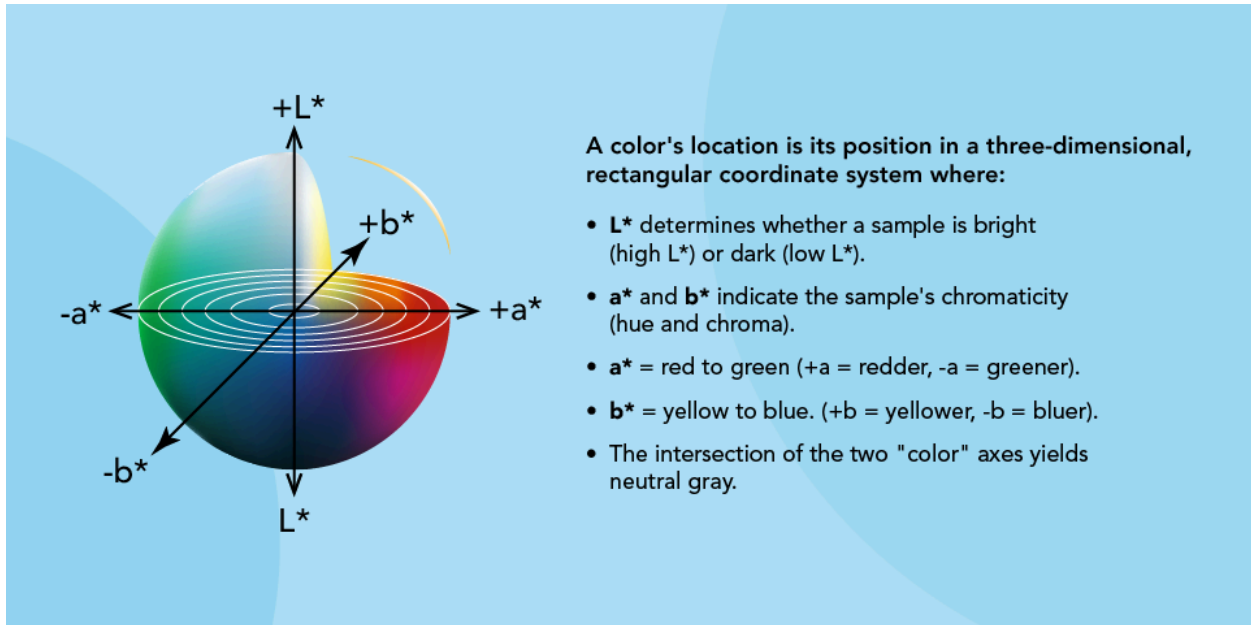


Figure (2.3.2) CIELAB Color Space from Hunter Labs [6]

2.3.2 Database Design

Overview of Database Design

The ER (Entity - Relationship) Diagram illustrates the relationship between the Brands, Products, and Shades that we are comparing with the user's measured values. This design utilizes multiple tables to optimize data storage and retrieval.

Entity Relationships:

(1:1) Shades → CIELAB: Shades and CIELAB have a one-to-one relationship to one another. The tables are linked such that each record in each table only appears once. The RGB and LAB values associated with a shade are located in the CIELAB table where Shade ID is a foreign key referencing the unique shade in the shades table.

(1:N): Brands → Products: Brands and Products have a one-to-many relationship, a brand can have multiple products but a singular product is associated with only one Brand. The products are referenced to a brand via the Brand ID foreign key.

(1:N): Products → Shades: Products and Shades have the same relationship as Brands and Products. A product can have multiple shades of a singular item but a shade is associated with one product. Shades are linked to products via the Product ID foreign key.

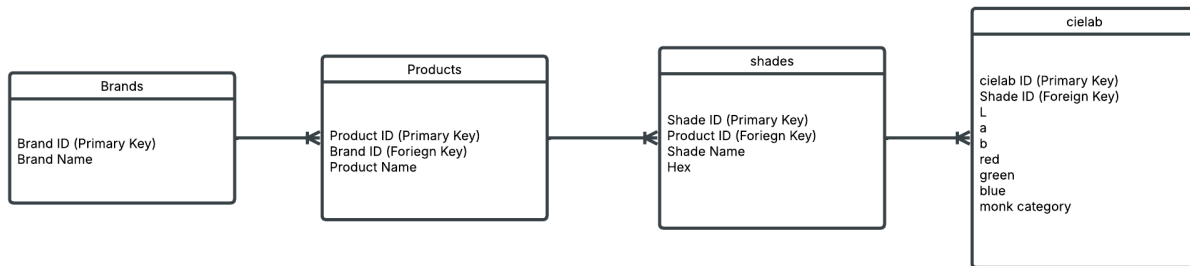


Figure (2.3.3) Database Design for Reference

Optimized Data Structure

There are several benefits to separating the data into multiple tables:

- **Reduced Redundancy:** Storing brands and product information separately eliminates the need to repeat the attribute for each shade
- **Improved Data Integrity:** Relationships between tables ensure consistency and reduce the risk of data anomalies and noisy data.
- **Enhanced Querying:** The above structure allows for more efficient and flexible queries when filtering by a specific attribute (Monk Skin Tone Category)
- **Scalability:** As the database grows, the structure can accommodate new data more easily without requiring reconstruction. As brands launch more products, we can update the products, shades, and CIELAB table and use the corresponding brand ID.

The CIELAB table includes the monk category for corresponding shades which can quickly filter shades based on the user’s classification, reducing the initial dataset for comparison.

As evident from Figure (2.3.3), the distribution of shade data across different categories is not uniform. Some categories contain a significantly larger number of shades compared to others, especially darker shades. This uneven distribution presents an opportunity to optimize our querying and comparison process.

By initially filtering the shades based on LAB values before calculating the ΔE between the user’s input and the database shades, we can improve the efficiency of our matching algorithm, particularly for categories with sparse data. For the darker categories of the Monk scale, filtering by LAB values dramatically decreases the number of ΔE calculations required. This is beneficial in minimizing unnecessary computations for underrepresented categories.

Justification

We aim to ensure fairness and reduce bias by creating a database that incorporates a wide range of foundation shades from different brands and is explicitly selected to represent the diversity of skin tones across the Skin Tone Scales. We acknowledge the beauty industry's underrepresentation of darker skin tones and are working to address this gap in our database. We will quantitatively assess this by tracking the number of shades within each category. Our goal is to achieve a distribution of products that reflects the real-world distribution of skin tones in the beauty industry.

Algorithm Design & Validation

We have adopted the Monk Skin Tone Scale to classify skin tones. It provides a more granular and inclusive representation of skin tones, especially for darker complexions. Our device will utilize the CIELAB color space, which is perceptually uniform and designed to approximate human vision. This helps to minimize color distortions and ensures accurate skin tone representation.

2.4 Tolerance Analysis

2.4.1 XYZ to L*a*b* Conversion

In comparison to RGB, CIELAB is more accurate in representing how humans perceive color differences primarily with two skin tones. The distance in the RGB space is not as linear in comparison to CIELAB.

To convert from RGB to XYZ color space, we would require a gamma expansion (linearization of RGB values) and matrix multiplication, the conversion is dependent on the specific RGB color space and reference white point. While to convert from XYZ to Lab* we use the following 3 formulas [9]:

$$F(t) = \begin{cases} 116t^{1/3} - 16 & \text{if } t > (6/29)^3 \\ \left(\frac{29}{3}\right)^3 t & \text{otherwise} \end{cases}$$

$$L^* = F\left(\frac{Y}{Y_n}\right)$$

$$a^* = \frac{500}{116} \left[F\left(\frac{X}{X_n}\right) - F\left(\frac{Y}{Y_n}\right) \right]$$

$$b^* = \frac{200}{116} \left[F\left(\frac{Y}{Y_n}\right) - F\left(\frac{Z}{Z_n}\right) \right]$$

X_n, Y_n, Z_n represents the white points.

Directly using XYZ is more sensitive to color variations provides a more accurate color representation and is not affected by device-specific variations compared to the RGB sensor. The calculations above, while extensive, can be simplified using the MATLAB function `xzy2lab()` [10].

The reference white points represent a standard illumination value in Figure (2.4.1)

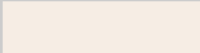
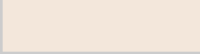
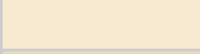




Value	White Point
"a"	CIE standard illuminant A, [1.0985, 1.0000, 0.3558]. Simulates typical, domestic, tungsten-filament lighting with correlated color temperature of 2856 K.
"c"	CIE standard illuminant C, [0.9807, 1.0000, 1.1822]. Simulates average or north sky daylight with correlated color temperature of 6774 K. Deprecated by CIE.
"e"	Equal-energy radiator, [1.000, 1.000, 1.000]. Useful as a theoretical reference.
"d50"	CIE standard illuminant D50, [0.9642, 1.0000, 0.8251]. Simulates warm daylight at sunrise or sunset with correlated color temperature of 5003 K. Also known as <i>horizon light</i> .
"d55"	CIE standard illuminant D55, [0.9568, 1.0000, 0.9214]. Simulates mid-morning or mid-afternoon daylight with correlated color temperature of 5500 K.
"d65"	CIE standard illuminant D65, [0.9504, 1.0000, 1.0888]. Simulates noon daylight with correlated color temperature of 6504 K.
"icc"	Profile Connection Space (PCS) illuminant used in ICC profiles. Approximation of [0.9642, 1.000, 0.8249] using fixed-point, signed, 32-bit numbers with 16 fractional bits. Actual value: [31595,32768, 27030]/32768.

Figure (2.4.1) From MATLAB xyz2lab resource.

2.4.2 Monk Skin Tone Scale Classification

To ensure consistency in classifying both the user's skin tones and the database shades, we employ a uniform calculation method. Our approach utilizes the Monk Skin Tone (MST) Scale, developed by Dr. Ellis Monk in collaboration with Google [12]. The MST Scale provides a more inclusive and representative range of skin tones compared to the previous classification systems (Fitzpatrick scale).

For quantitative measurements and precise color representations, we will use the MST Google AI Skin Tone Research Hex Values. These standardized color values are specifically designed for research purposes, enabling accurate studies and evaluations of skin tone diversity. By adopting these exact color values we ensure consistent classification across user inputs and database entries and align with current computer vision research standards.

Monk Scale	Hex Color	L*	a*	b*
1		94.211	1.503	5.422
2		92.275	2.071	7.28
3		93.091	0.216	14.205
4		87.573	0.459	17.748
5		77.902	3.471	23.136
6		55.142	7.783	26.74
7		42.47	12.325	20.53

8		30.678	11.667	13.335
9		21.069	2.69	5.964
10		14.61	1.482	3.525

Figure (2.4.2) Monk Scale Conversion in L*a*b* Format

CIELAB* is a 3D color space based on the Opponent-Color Theory which assumes that the receptors in the human eye perceive color as the following pairs [7]:

- **L Light vs Dark:** 0 - 50 indicates dark while 51 - 100 indicates light
- **a Red vs Green:** A positive number indicates red, negative number indicates green
- **b Yellow vs Blue:** A positive number indicates yellow and a negative number indicates blue

*Note: CIELAB has no defined units and is interpreted based on range.

To quantify the difference between specific colors within the color space, we utilize ΔE^* which is an equation that describes how perceptually different two colors are [13]. ΔE^* measures the total color difference and represents the distance between a sample and a standard color. This metric is valuable for determining an acceptable tolerance range for color differences.

A ΔE^* of 0.00 indicates that the two colors are identical and a $\Delta E^* < 2$ represents a perceptibly small difference. For this project, we are using a threshold $\Delta E^* \leq 1.75$ that would allow us to further reduce and optimize our querying as we remove any shades that are dissimilar to our measured values.

Equation: (Note ΔE^* has no units)

$$\Delta E^* = \sqrt{(\Delta L)^2 + (\Delta a)^2 + (\Delta b)^2}$$

This formula calculates the Euclidean distance between two LAB values in the color space. To classify our shades into one of the 10 Monk categories, we will calculate the ΔE^* (Euclidean distance) between each shade's LAB value and the LAB value of the 10 Monk categories shown in Figure (2.4.2). While this process is computationally intensive, it only needs to be performed once to appropriately classify each individual shade. We assign each shade to the category with the smallest ΔE^* value.

This approach allows for a more precise and inclusive categorization of skin tones, aligning with the Monk Skin Tone Scale's goal of improved representation across diverse skin colors.

We will apply the same steps to the LAB values measured via our color subsystem to the 10 LAB values of the Monk scale. Once the measured values are classified, we can apply a filtered query to the CIELAB table to find all the shades within the same class as the user.

2.4.3 Product Matching & Verification (Testing)

To validate the accuracy of our product-matching calculations, we leverage K-Means clustering for visual confirmation. While K-Nearest Neighbors (KNN) and K-Means clustering are computationally intensive and sensitive to outliers when used for real-time shade matching (due to their reliance on the entire

database), they are valuable tools for visually confirming our classification methodology. After classifying the user's measured CIELAB values into one of the 10 Monk categories, we performed a K-Means clustering analysis (with $k=10$) on a set of manual data to represent the Monk categories.

Testing Procedure

For the purpose of data testing (prior to PCB integration), we verified the accuracy of our color-matching algorithm against the database using the following steps (implemented in `multi_sensors.ipynb`):

1. Data Acquisition

- a. Multiple images of a single face were captured under varying lighting conditions.
- b. Using Procreate, several points on the face were color-picked to obtain the HEX values of the skin in each image.

2. CIELAB Conversion

- a. The acquired HEX values were converted to CIELAB values using `colorhexa.com`.
- b. Two arrays were created: `target_hex` (containing the HEX values) and `target_cielab` (containing the corresponding CIELAB values).

3. Data Visualization

- a. The measured colors were displayed using Matplotlib for visual inspection. (**Figure 2.4.3.1**)

4. Average CIELAB Calculation

- a. The average Lab* values of the measured data points were calculated using `np.mean` on the `target_cielab` array.
- b. The calculated average CIELAB value was displayed (**Figure 2.4.3.2**)

5. Color Difference Analysis

- a. The color difference ($\Delta E^* \leq 1.75$) was calculated between each measured `target_cielab` value and the calculated average CIELAB value.
- b. Measured `target_cielab` values that met the ΔE^* threshold requirements were saved to an array.

6. K - Means Clustering

- a. K-Means clustering was performed on the database and the valid data points (obtained in step 5) for visualization and product matching purposes. (**Figure 2.4.3.3**)

7. Match Visualization

- a. The color difference (ΔE^*) values were displayed, along with the data points that were within the acceptable threshold of the target-measured CIELAB value. (**Figure 2.4.3.4**)*
(**Note*** Color Bar is opposite to the graph, purple indicates a larger Delta E)
- b. The top 10 matches to the user (where $\Delta E^* \leq 1.75$) were displayed for visual confirmation. (**Figure 2.4.3.5**)

8. Product Information Retrieval

- a. Using the `shade_id` foreign key associated with the matching shades, queries were performed on the Brands, Products, and Shades tables to retrieve corresponding product information.
- b. The retrieved product information was printed to the terminal. (**Figure 2.4.3.6**)

This procedure enables us to validate the performance of our color-matching algorithm against the database and assess its accuracy in identifying appropriate product recommendations without full integration of our PCB.

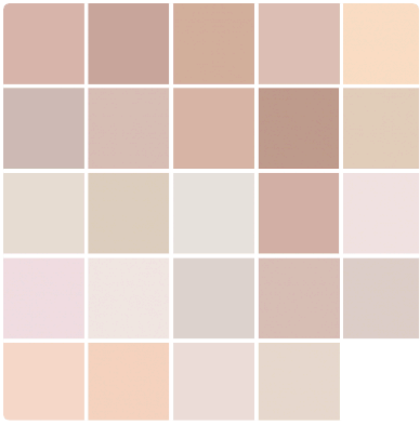


Figure 2.4.3.1

Average CIELAB Value



Figure 2.4.3.2

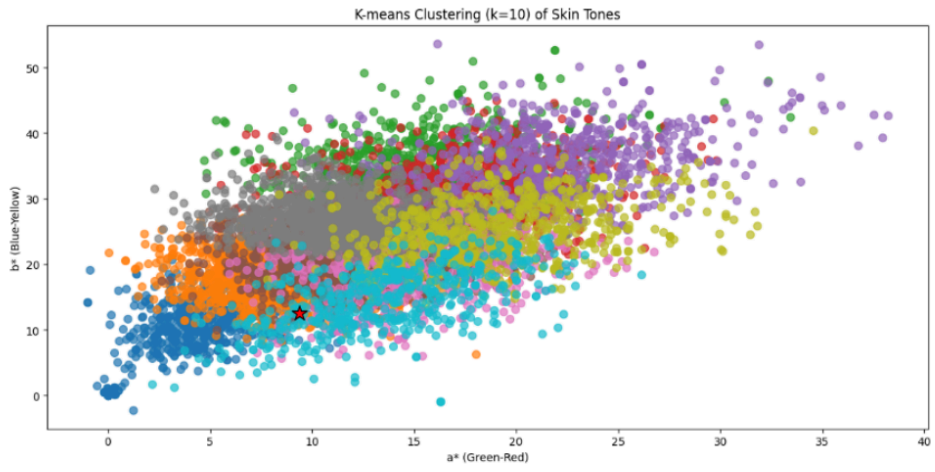


Figure 2.4.3.3

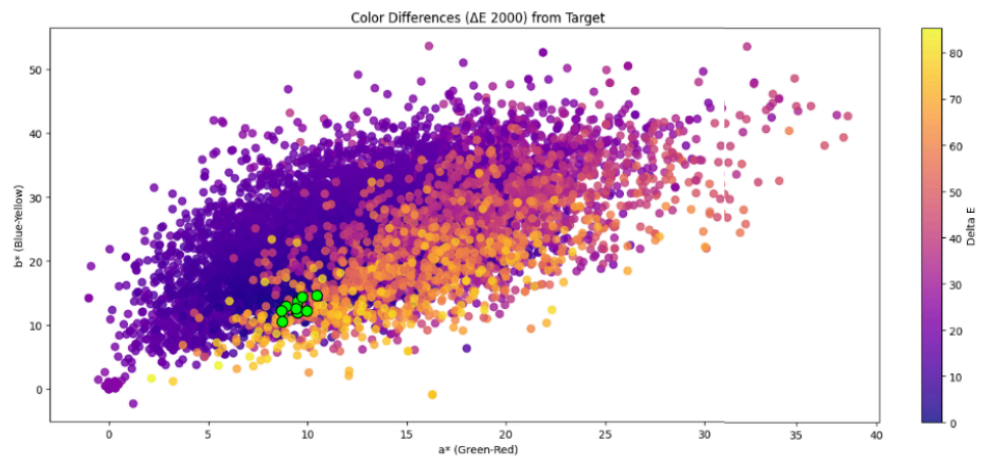


Figure 2.4.3.4

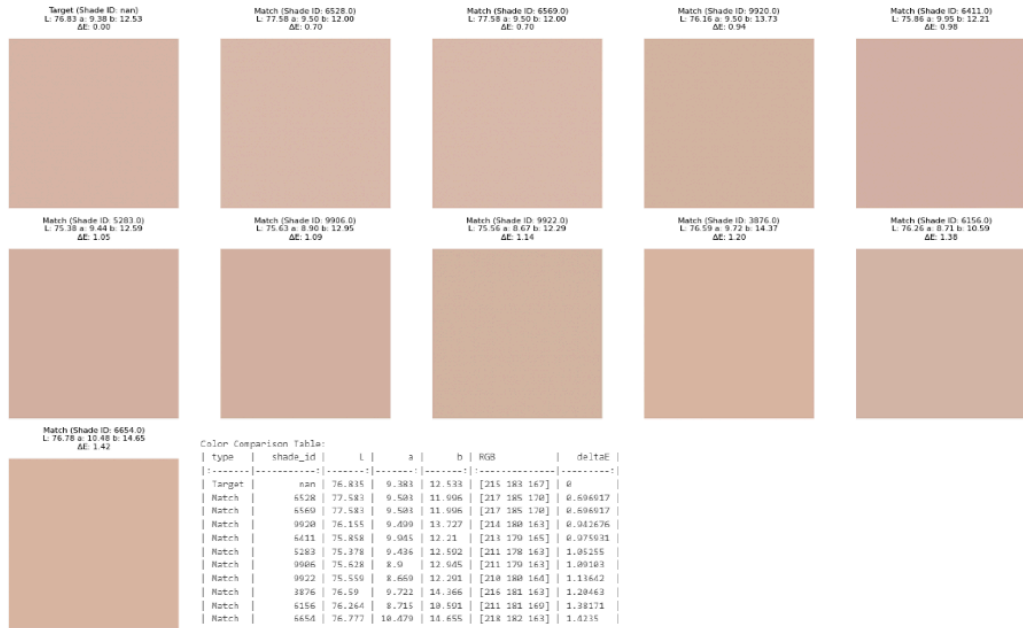


Figure 2.4.3.5

Brand	Product	Shade	Delta E
KVD Beauty	Good Apple Skin-Perfecting Foundation Balm	Light 006	0.696917
KVD Beauty	Good Apple Full-Coverage Transfer-Proof Serum Foundation	Light 006	0.696917
SEPHORA COLLECTION	Matte Perfection Powder Foundation	04 porcelain pink	0.942676
KEVYN AUCOIN	The Etherealist Foundation	01 light ef	0.975931
Guerlain	Lingerie De Peau Natural Perfection Foundation SPF 20	02C light cool	1.05255
SEPHORA COLLECTION	10 Hour Wear Perfection Foundation	17.5 oat	1.09183
SEPHORA COLLECTION	Matte Perfection Powder Foundation	08 fair neutral	1.13642
ColourPop	Pretty Fresh Foundation	83C medium	1.20463
Jouer Cosmetics	Essential High Coverage Crème Foundation	vanilla	1.38171
L'Oréal	True Match Super-Blendable Foundation Makeup	alabaster	1.4235

Figure 2.4.3.5

The Shade matches were then verified manually by comparing the corresponding products and shades in person at the store.

2.4.4 Undertone Analysis

The ΔE^* calculation inherently loses the directional information needed to determine undertones. This is because ΔE^* involves squaring the differences in L^* , a^* , and b^* values, discarding the sign of each difference. While ΔE^* is useful for quantifying the magnitude of overall color difference, it cannot reveal whether the difference is due to a dominance of red, blue, green, or yellow – information essential for undertone analysis.

Therefore, to properly understand undertones, we aim to analyze the a^* and b^* values separately to determine underlying color biases. Since there is no standardized measurement or established methodology for directly measuring undertones via $L^*a^*b^*$ values and comparing accuracy, we can estimate the most probable category a user's skin tone falls under by comparing the measured a^* and b^*

values. This approach allows us to infer the undertone category until a reliable method and supporting research for determining undertones is established [10].

We will estimate the most probable undertone category for a user’s skin by comparing the measured a^* and b^* values against the following criteria:

- **Warm Undertone:** $+ a^*$, $a^* > b^*$ and $+ b^*$
- **Cool Undertone:** $- a^*$, $a^* < b^*$ and $- b^*$

Using the user’s measured $L^*a^*b^*$ values, we will compare the a^* and b^* channels using the above comparison metrics to display the undertone category they most likely fall into.

3.1 Cost

Item Description	Manufacturer	Link	Part Number	Retail Price	Our Cost	Quantity	Total Cost
ESP32-S2-WROOM-1	Espressif	Mouser	ESP32-S3-Dev KitC-1-N8R2	\$15	\$0	1	\$0
ESP32-S2-WROOM	Espressif	DigiKey	ESP32-S3	\$3.56	\$3.56	1	\$3.56
Proximity, Light, RGB, and Gesture Sensor - STEMMMA QT / Qwiic	Adafruit	Adafruit	APDS9960	\$7.50	\$7.50	1	\$7.50
2000mAh 7.4 V 2S Rechargeable RC Battery	URGENEX	Amazon	7.4 V Li-ion Battery	\$25	\$25	2	\$25
LDO Voltage Regulator AZ1117CH - 3.3TRG1	Diodes Incorporated	DigiKey	AZ1117CH	\$2.55	\$2.55	10	\$2.55
XYZ Color Sensor	TI	Mouser	OPT4048	\$2.36	\$2.36	3	\$7.08
RGB Sensor Breakout	Sparkfun	Mouser	OPT4048	\$10	\$10	1	\$10
GPIO Female Pin Headers	Treedix Store	Amazon	N/A	\$7.59	\$7.59	1	\$7.59

USB Connector	GCT Better Connected	Mouser	40-USB3140300170C	\$0.83	\$0.83	1	\$0.83
USB to UART Bridge	Silicon Labs	Digikey	CP2104-F03-GMR	\$6.47	\$6.47	1	\$6.47
RGB LED	Supply Center	ECE	Blue Cell Multicolor LED	\$0.5	\$0.5	3	\$1.5
TVS Diode	Littelfuse Inc.	Digikey	SP0503BAHTG	\$0.61	\$0.61	1	\$0.61
WHITE LED	E-Shop	Student Self Help	N/A	\$0	\$0	2	\$0
LTST-C191KFKT	E-Shop	Student Self Help	N/A	\$0	\$0	2	\$0
LTST-C190KSKT	E-Shop	Student Self Help	N/A	\$0	\$0	2	\$0

Name	Hourly Rate	Hours Invested	Total
Ashley Herce	\$50	50	\$6,250
Waidat Bada	\$50	50	\$6,250
Shriya Surti	\$50	50	\$6,250
Total			\$18,750

$$\text{Labor Cost} = \text{Hourly Rate} \times \text{Actual Hours Spent} \times 2.5$$

Section	Total
Labor	\$18,750
Parts	\$72.69
Total	\$18,822.69

3.2 Schedule

Week	Task	Responsibility
2/24	PCB Review Due	All
	Create a makeup database using SQLite	Ashley
	Contact brands for makeup samples for testing	Ashley
	Research hardware modules and design PCB design	All
	Complete PCB KiCad for First Round PCBway Order	All
	Work on Design Document	All
	Purchase hardware & all parts	All
3/3	First Round PCBway Orders Due (3/3)	All (Shriya)
	Teamwork Evaluation I (3/5)	All
	Design Document Due (3/6)	All
	Breadboard Demo with Instructor & TA (3/10)	All
	Test functionality of the RGB Light Breakout with ESP32 Dev Board	All
	Develop Python script to run queries on database	Ashley
	Write a program to connect to ESP32-S2 Development Board BLE	All
	Test sensor and Power Subsystems	Shriya & Waidat
	Work on connecting BLE with ESP32 Dev board for breadboard demo	Ashley & Shriya
3/10	Second Round PCBway Order (3/13)	Ashley & Shriya
	Work on PCB redesign with any modifications and additional parts that were missed during PCB order 1	All
3/17	Work on cleaning up Database Data and continue to web-scrape brand links for information	Ashley
	Create Mock Up and detailed steps and research for UI design	Ashley
3/24	Design Swift UI for mobile application	Ashley & Waidat
	Implement CRUD on UI applications with a database	Ashley

	Connect mobile application with microcontroller BLE	All
	Test BLE connection with sensors and reading data	All
3/31	Third Round PCBway Order (3/31)	All (Shriya)
	Individual Progress Report Due	All
	Work on Individual Progress Reports	All
	Run tests on the final project and display sensor data to the UI	All
	Debug any issues and bugs with the design	All
4/7	Fourth Round PCBway Order (4/7)	All
4/21	Mock Demo	All
4/28	Final Demo/Mock Presentations	All
5/5	Final Presentation	All

Section 4: Ethics & Safety

4.1 Ethics

4.1.1 Algorithmic Fairness & Bias Mitigation

Our team is dedicated to upholding ethical standards and ensuring that our skin tone analysis and product matching algorithms are free from bias and treat all users equitably. We recognize the potential for algorithmic bias to perpetuate existing inequalities, and we are committed to mitigating this risk through careful design and validation.

In alignment with current AI and computer vision research standards and ACM Code of Ethics 2.7, we are employing the Monk Skin Tone Scale, developed by Dr. Ellis Monk in collaboration with Google, to classify skin tones. This scale provides a more inclusive and representative range of skin tones compared to the Fitzpatrick scale and ensures that our algorithms are not inadvertently biased against darker skin tones which we found in our data, a common issue within the beauty industry that we hope to address through our product.

Our algorithms will be implemented using standard illuminants, which will provide a basis for comparing colors that are recorded under different lighting conditions. It's important to note that Google advises against equating the shades in the Monk Skin Tone Scale with race, as skin tones can vary within race, and the skin tone analysis is solely intended for cosmetic product matching and should not be used to make inferences about a user's race or ethnicity.

4.1.2 Environmental Responsibility

Consistent with ACM Code 1.1 (public good) [12], our design and manufacturing processes aim to minimize environmental impact. This includes selecting sustainable materials where possible, designing for energy efficiency, and adhering to e-waste recycling guidelines for all components, including batteries and electronic parts.

4.2 Safety

4.2.1 Battery Safety

The 7.4V Li-ion battery pack used in our device complies with UL 2054 standards, incorporating overcurrent protection and FCC Part 15 EMI shielding to prevent potential hazards. However the LDO may risk dispatching more power than it is rated for, so we will use a heat sink for protection. This aligns with IEEE 1725 standards for rechargeable battery safety.

4.2.2 Disposal

To minimize environmental impact, we will adhere to campus e-waste disposal guidelines for recycling PCBs and LEDs through the UIUC Sustainable Electronics Center.

4.2.3 Laboratory Safety Compliance

Development and testing will be conducted in University of Illinois laboratories, where full compliance with campus safety policies is essential. This involves following electrical safety procedures, handling PCB components with care, and observing all lab-specific regulations to maintain a safe and hazard-free workspace.

Section 5 References

- [1] P. Goon, C. Banfield, O. Bello, and N. J. Levell, "Skin cancers in skin types IV–VI: Does the Fitzpatrick scale give a false sense of security?," *Skin Health and Disease*, vol. 1, no. 3, Jun. 2021, doi: <https://doi.org/10.1002/ski2.40>.
- [2] HTML Color Codes, "HTML Color Chart," Available: <https://htmlcolorcodes.com/color-chart/>.
- [3] Bold Hue, "Bold Hue – Find Your Perfect Color Match," Available: <https://www.boldhue.com/>.
- [4] Texas Instruments. "Choosing Between RGB and XYZ Color Sensors for Adaptive Lighting Adjustments." Accessed: Mar. 06, 2025. [Online]. Available: https://www.ti.com/lit/ab/sboa567/sboa567.pdf?ts=1740975775684&ref_url=https%253A%252F%252Fwww.google.com.mx%252F
- [5] C. Heldreth, E. P. Monk, A. T. Clark, S. Ricco, C. Schumann, and Xango Eye, "Which Skin Tone Measures are the Most Inclusive? An Investigation of Skin Tone Measures for Artificial Intelligence.," *ACM Journal on Responsible Computing*, Nov. 2023, doi: <https://doi.org/10.1145/3632120>.
- [6] Wikipedia Contributors, "Fitzpatrick scale," *Wikipedia*, Oct. 22, 2019. https://en.wikipedia.org/wiki/Fitzpatrick_scale
- [7] P. Ken. HZDG, "What Is CIELAB color space?," *Hunterlab.com*, 2022. <https://www.hunterlab.com/blog/what-is-cielab-color-space/>
- [8] "Convert XYZ Color to L*a*b*," *Mathworks.com*, 2025. <https://www.mathworks.com/help/images/ref/xyz2lab.html> (accessed Mar. 06, 2025).
- [9] T. Fujiwara. "Color space conversion (3)," *Sakura.ne.jp*, 2019. https://fujiwaratko.sakura.ne.jp/infosci/colorspace/colorspace3_e.html (accessed Mar. 06, 2025).
- [10] openoximetry.org, "Skin Color Quantification - OpenOximetry," *OpenOximetry*, Sep. 13, 2024. <https://openoximetry.org/skin-color-quantification/>
- [11] "Sensor Instruments," *www.sensorinstruments.de*. <https://www.sensorinstruments.de/whatiswhat.php?subpage=11&language=en>
- [12] M. Ellis. "Skin Tone Research @ Google," *skintone.google*. <https://skintone.google/get-started>
- [13] "Brief Explanation of delta E or delta E*," *Hunterlab*, Jul. 20, 2022. <https://support.hunterlab.com/hc/en-us/articles/203023559-Brief-Explanation-of-delta-E-or-delta-E>
- [14] Association for Computing Machinery, "ACM code of ethics and professional conduct". [Online]. Available: <https://www.acm.org/code-of-ethics>. [Accessed: Feb. 13, 2025].
- [15] Institute of Electrical and Electronics Engineers. "IEEE Code of Ethics". [Online]. Available: <https://www.ieee.org/about/corporate/governance/p7-8.html> [Accessed: Feb. 13, 2025]