CLASSROOM CLARITY SUPPORT HUB

By

Maddie Donku
Kaitlin Gowens
Jesse Gruber

Final Report for ECE 445, Senior Design, Spring 2025
TA: Aishee Mondal
Professor: Michael Oelze

7 May 2025
Project No. 21

Abstract

This document provides a detailed description of the ECE 445, Senior Design project, the ClassroomClarity Support Hub. The support hub combines a central hub and an app to facilitate communication between students and instructors through light-emitting diodes (LEDs) and a screen display to improve the learning environment. The ClassroomClarity is divided into four subsystems: power management, controls, feedback, and mobile app subsystem. Each subsystem was designed and tested against a variety of functionality requirements which all showed successful results, therefore presenting a fully functioning final project.

Contents

1. Introduction

ClassroomClarity is a device created to facilitate communication between students and instructors in a live lecture setting. The following section provides insight into the problems ClassroomClarity addresses and how it solves them. From there, the report dives into the specific subsystems that make up ClassroomClarity and how they function. Cost, timing, ethics, and future work are also discussed.

1.1 Problem

Within a classroom setting, communication between students and teachers is key for the optimal learning experience. However, there are several common barriers that can disrupt this interaction. Firstly, when going through difficult material, instructors may go too fast causing students to become confused or go too slow causing students to lose focus. Secondly, many students today are too anxious to raise their hand in class. Questions are crucial for the learning process, not only because they help clear confusion, but also because they allow students to actively participate. Finally, many instructors become very focused on their presentation materials during a lecture which leads to raised hands going unnoticed. After waiting too long to be called on, students may decide not to ask their question if the instructor has already moved past the material the question was about, or they are tired of waiting.

While there are some online tools, like Mentimeter, that try to resolve some of these problems, none of them provide full, seamless support. Many of these platforms require the instructor to create extra slides outside of class and they also take up space on the lectern. They also do not provide a means of reminding the professor to check for questions. Professors and students can benefit from a tool that will easily show them how the class is handling material and any questions that may arise. A hub that is consistent between classes will simplify the learning experience for both students and professors.

1.2 Solution

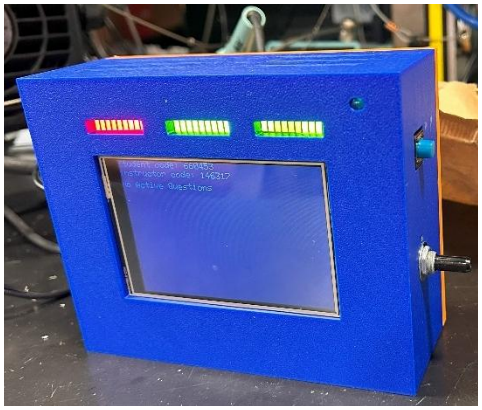


Figure 1: Image of the instructor's portable hub

ClassroomClarity strives to alleviate the common classroom barriers through a system consisting of a physical hub for the instructor and a mobile app for the students. The hub, as shown in Figure 1, connects to the app through Bluetooth. It includes three LED arrays, red, yellow, and green, which indicate to the professor the students' current level of understanding. The number of LEDs lit will guide the professor on whether to speed up or slow down as less LEDs lit means less students understand. Students can edit their level of understanding in the app by pressing one of the five buttons labeled 1-5, with one representing the least understanding. The hub also has a screen which displays the submitted questions. Students can submit questions via the app in three forms. First, they can press the "raise-hand" button which submits their display name to the question queue to inform the instructor they have a live question. Second, they can type out and submit a question which will be displayed on the screen along with their display name. Finally, students can choose to submit anonymously if they are feeling shy. When any version of a question is submitted, a blue, notification LED turns on and the vibration motor within the hub vibrates to notify the instructor. The instructor can stroll through the questions in the queue using the dial, as well as clear questions as they answer them by pressing the button. The notification LED will remain on, and the vibration motor will go off every two minutes until all questions are cleared. If the professor doesn't want the vibration notification, they can log into the instructor side of the app and turn it off along with changing the font size of the screen. To access the student or instructor side of the app, the user must input the corresponding password that is displayed on the hub's screen.

1.3 Block Diagram

Figure 2 illustrates the various subsystems that make up the hub. These subsystems include power management, control, and feedback subsystems and the mobile app. The power management subsystem converts 5 VDC to 3.3 VDC that is used by the other subsystems. The control system operates the output signals for the feedback system depending on input signals from the button, dial, and mobile app. Finally, the feedback system displays the received data to the user. The original block diagram differed from the final because now all the components run on 3.3 V instead of just the rotary encoder being run on 5 V, so it better communicates with the microcontroller.

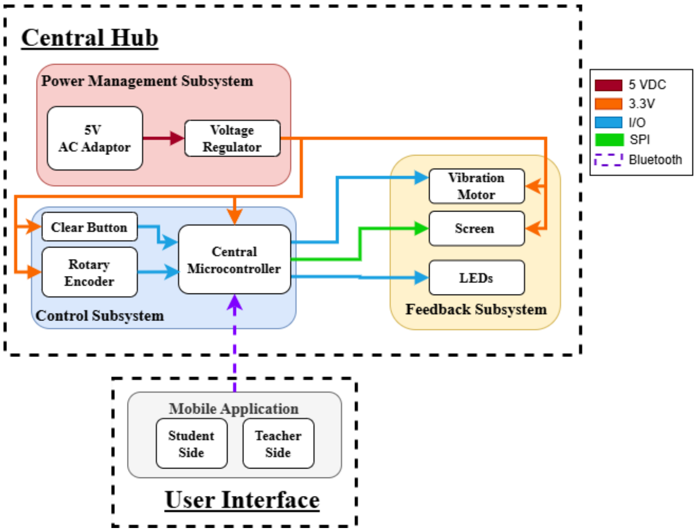


Figure 2: Block diagram of ClassroomClarity system

1.4 High Level Requirements

To deem the Classroom Clarity Hub a success, the following requirements must be met:

1. Questions can be sent from the app to the hub's question queue, which should be able to hold at least 5 questions, such that at least 1 question is displayed by the hub if the queue is not empty.
2. When a question is received by the hub, the hub should vibrate for 1 second and light the indicator LED within 30 seconds of receiving it. The LEDs turn off within 30 seconds of the question queue becoming empty.
3. Students can select their understanding level on the app and the LEDs on the hub should change within 30 seconds of the selection to match the new average level of understanding.

2. Design

2.1 Power Management Subsystem

The power management subsystem is responsible for delivering power to the hub's internal subsystems in a reliable and safe manner. Figure 3 walks through the flow of power from the primary supply to the power recipients. The primary supply that was selected for the creation of this project was the AC wall outlet for its stable and available power levels. The wall outlet provides 120 VAC to an adapter that converts the voltage to AC and steps it down to 5 VDC. That 5 V from the adapter is then passed to a DC/DC converter where it is stepped down to 3.3 V and passed to the rest of the subsystems.

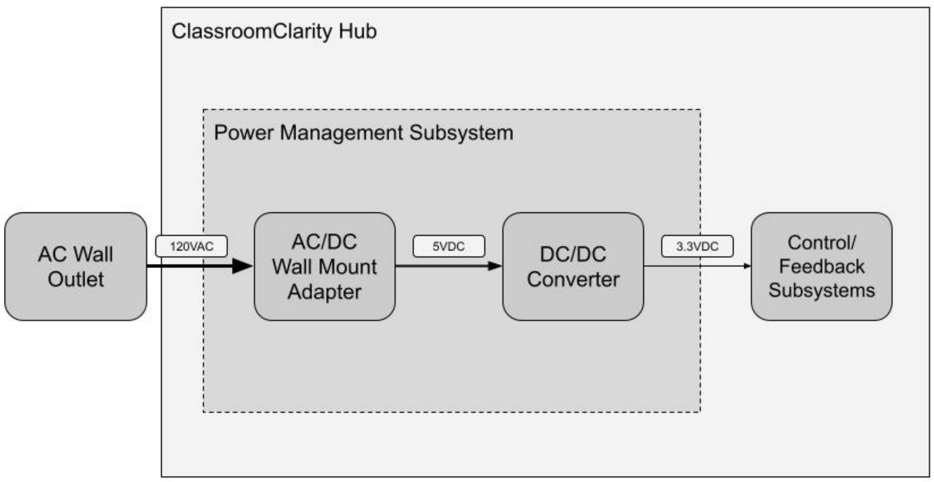


Figure 3: Block diagram of Power Management Subsystem

To choose the correct components to deliver power to the hub, some analysis and calculations had to first take place. Table 1 gives the relevant voltage and current requirements of the primary subsystem components. These components are those that have the heaviest power needs and are integral to the hub's function. Looking at the voltage needs of each component, a maximum of 5V would be needed to be supplied to the hub. This fits nicely with market components, since 120 VAC to 5 VDC adapters are commonly available.

Table 1: Voltage requirements for Hub components

Component	Peak Voltage	Peak Current
Microcontroller (ESP32-S3-WROOM-1) [1]	3.3 V	500 mA
TFT LCD (MSP4022) [2]	5 V	100 mA-500 mA
Vibration Motor (316040004) [3]	3.5 V	104 mA
LED – Blue [4]	4 V	20 mA

To get an estimate of the total current that would be drawn by the PCB, Equations 1-3 were used. The maximum current for each major component was collected from their respective data sheets and added together to get the minimum current that the power management subsystem must provide. For the LED estimation, the color of LED that takes the most amount of current (blue) was assumed to be the color of every LED for simplicity in calculation and to leave room for error if more current was needed down the line. The PCB draws as much current as it needs from the supply so having a higher current rating than needed was not an issue, but standard practice.

$$I_{total} = I_{microcontroller} + I_{screen} + I_{motor} + I_{LEDs} \tag{1}$$

$$I_{total} = 500\text{ mA} + 500\text{ mA} + 104\text{ mA} + 19\times 20\text{ mA} \tag{2}$$

$$I_{total} = 1.484\text{ A} \tag{3}$$

With the final current value of approximately 1.5 A, the constraints for the final converter were established and the component model could be selected. Initially, a low dropout (LDO) regulator was examined but did not have models that could handle the current needed for the project. After some research on affordable power conversion models, the TPSM84203EAB buck converter module was selected. The module has a fixed output of 3.3 V and the voltage and current specifications are displayed in Table 2. The buck converter module was the ideal component due to its small size, affordability, and efficiency.

Table 2: Maximum rating for DC/DC Converter

Component	Max Voltage	Max Current
DC/DC Converter (TPSM84203EAB) [5]	30 V	1.5 A

For the final physical design of the power management subsystem, all components from the block design were brought together. A 5 V, 3 A wall mount adapter with a USB-C to DC cord was used to connect from the wall to the hub's PCB. The DC end of the cord connects to a barrel jack mounted on the PCB. The choice of a standard barrel jack allows for flexibility in the consumer's choice of power supply. Based on an individual consumer's needs, they may power the hub with alternate supplies such as a portable charger or battery module if they have the appropriate jack adapter. The connection from the barrel jack to the DC/DC converter module can be seen in Figure 4, displaying the KiCad schematic for the subsystem.

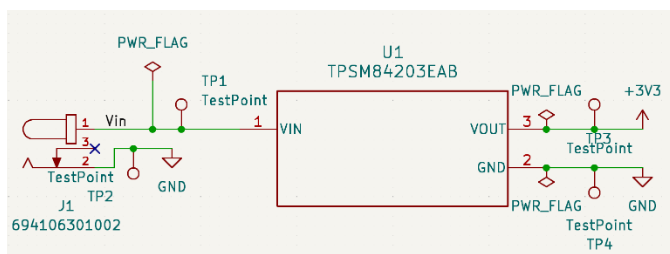


Figure 4: Power Management Subsystem Schematic

2.2 Control Subsystem

The control subsystem included the ESP32-S3-WROOM-1 microcontroller and physical interfaces to perform operations on the feedback subsystem (Figure 5). Between the physical interfaces and the feedback subsystem, the hub had many signals to read and write. This was a driving factor in choosing the ESP32-S3-WROOM-1 microcontroller as it has numerous GPIO pins to be used for the necessary signals. Furthermore, the vast amount of documentation on this microcontroller was another important point in the decision.

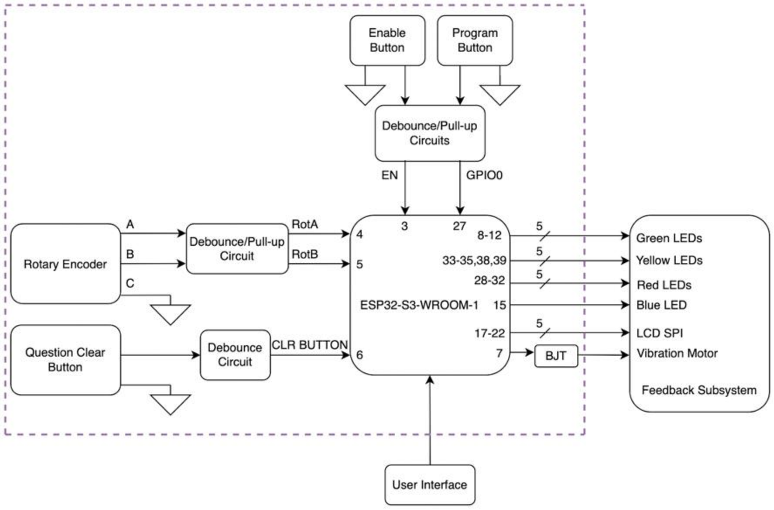


Figure 5: Block diagram of the Control Subsystem

The control system featured two buttons used internally to program the microcontroller and a button used to clear questions. Additionally, a rotary encoder was implemented to scroll through questions on the hub. As these physical parts have mechanical contacts, debouncing circuits were introduced to filter out noise in the signal. This was accomplished with RC circuits.

The circuits for the buttons are shown in Figure 6. The programming buttons, EN and GPIO0, must be pulled high to ensure the microcontroller will be powered and boot into execution mode. These strapping pins are set by Espressif and listed in Figure 20 (Appendix E). Additionally, the voltage of 3.3 V is the nominal high input voltage for the microcontroller. To be consistent, the question clear button was also pulled high, though there was no constraint whether this signal must be pulled high or low.

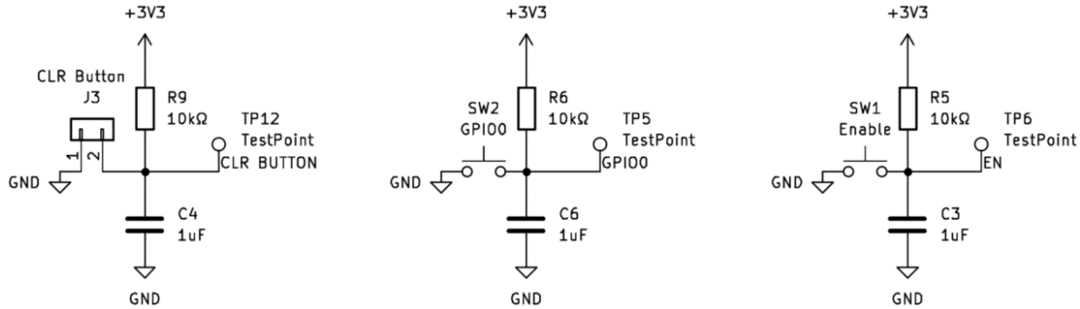


Figure 6: RC filter circuits for button debouncing

Likewise, the signals of the rotary encoder were pulled high, as seen in Figure 7. However, this was required as the rotary encoder uses quadrature encoding, and the encoder pulls the signals to low upon turning the dial. The direction the encoder is turned determines whether signal A or signal B leads the other, enabling the direction of rotation to be decoded by analyzing how the signals change in relation to each other.

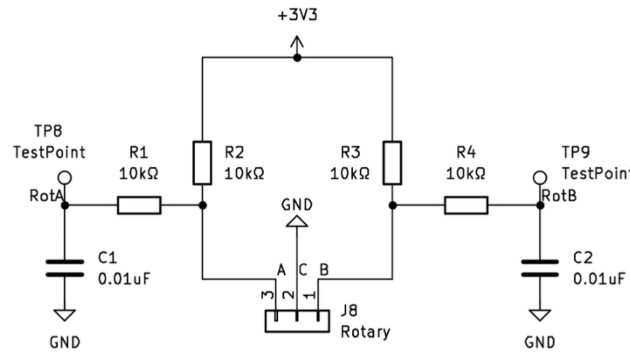


Figure 7: Rotary encoder debouncing circuit

To determine the resistance and capacitance for debouncing,

$$\tau = RC \tag{4}$$

was used to first find a baseline of 10 ms. This response was chosen based on a conservative time constant to filter the bounce of the button since its signal does not have to be of high quality. Using this baseline, a resistance of 10 kΩ and a capacitance of 1 µF was chosen. However, a more precise signal was desired for the rotary encoder as it may experience interactions at a higher rate than the buttons. The resistance was kept the same, but the capacitance was lowered to 0.1 µF to decrease the time response of the rotary encoder. While the capacitance of 0.1 µF was designed to be used, an error in communication at some point resulted in a capacitance of 0.01 µF to be used. The circuit was physically tested with this capacitance, and it behaved as intended.

To validate the time constants, simulations were completed in LTSpice using a 10kHz switching frequency and a duty ratio of 0.5. The switching frequency came from the worst-case scenario of a typical bounce occurring 10 to 100 times over 1 ms [6]. The results of the bounce-event simulations for the button and rotary encoder configurations are in Figures 12 and 13 (Appendix B).

The results showed that as the signals were "bouncing" from an unpressed to a pressed state, the voltage remained below the low input threshold of 0.8 V given by the ESP32 [1]. Also, the time to surpass the high input voltage threshold of 2.64 V from 0 V was observed. Seen in Figures 15 and 16, this threshold was crossed at 16.62 ms for the buttons and 3.73 ms for the rotary encoder (Appendix B). This smaller time response was desired for the rotary encoder to ensure robust processing as it may be rotated quickly. On the other hand, the larger time response was adequate for the buttons as their signals did not need to have a very high resolution due to an expected pause between physical interactions with the buttons.

For wireless communication between the central hub and the mobile app, Bluetooth Low Energy (BLE) was chosen over Wi-Fi, classic Bluetooth, and radio frequency (RF). While the ESP32 supports all four technologies, BLE is built-in to all modern smartphones, unlike RF, and offers low power operation, quick pairing, and doesn't require a full network to connect like Wi-Fi does [1]. BLE also has sufficient range to transmit over the average classroom size. Since the app doesn't send large data packets, BLE was the best choice for this project. When first initiated, the

ESP32 configures itself as a BLE device, creates a BLE server, sets up the BLE services and characteristics, and then begins advertising so the app can connect to it.

To efficiently manage the types of expected input data — understanding rating, question submission, setting change — each is assigned to its own characteristic with a unique UUID and read/write permissions. Additionally, the ESP32 initializes a separate characteristic for the password, enabling it to send whether it was correct or not to the app. The BLE functionality was implemented using the built-in Arduino library, Arduino BLE. This library was chosen over others, such as Bluetooth since those libraries work with classic Bluetooth and not BLE. It contains the essential subclasses such as BLEDevice, BLEUtils, BLEServer, and BLE2902, which handle discovering devices, service and characteristic creation, and notification functionality with the connected clients [7]. An overview of the full BLE setup and operation process is provided in Figure 18 (Appendix C).

With all parts of the control subsystem finalized, the method to process the signals was created. Shown in Figure 17 (Appendix C), the software went through each input signal and checked the state of the input. If a change was detected, action was taken to alter the behavior of the components on the feedback subsystem. For example, if the clear button was pressed and there was a question on the hub, the question would be deleted and removed from the display.

2.3 Feedback Subsystem

The feedback subsystem was the interface for the instructor to observe the status of their class (Figure 8). A 4-inch TFT LCD was used to display the student and instructor passcodes to access the app and questions sent in by the students. Also, a single blue LED was lit if there was a question on the hub, and three LED arrays indicated the average understanding level of the class. The arrays had ten LEDs on each module, and the LEDs were lit two at a time to fit the needed signals onto the limited GPIO pins of the ESP32, resulting in 15 sets of LEDs. If more students understood class material, more LEDs would be lit. An increase in confusion would result in less LEDs to be lit. Lastly, a vibration motor was featured as a clear indication that there was a question, and it would vibrate for 0.3 seconds every two minutes if a question remained on the hub as a reminder.

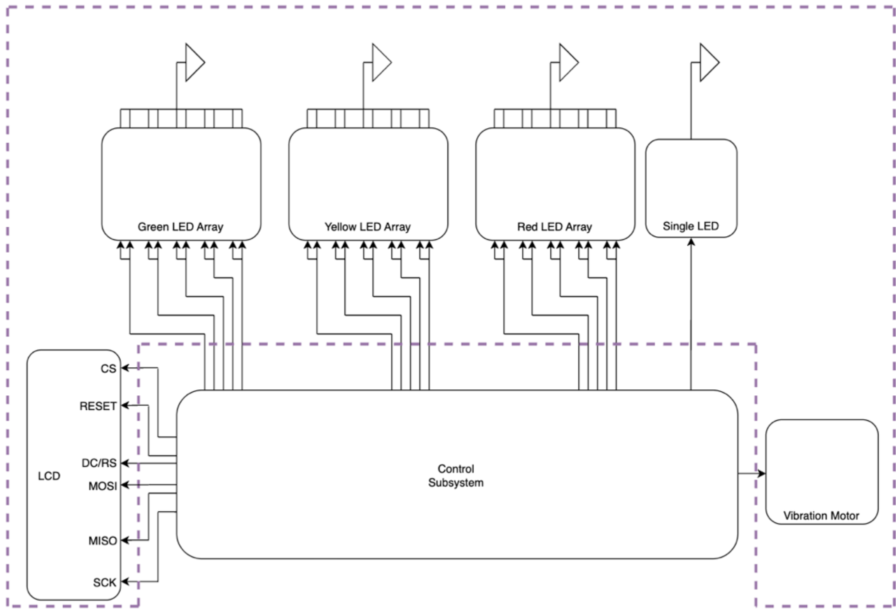


Figure 8: Block diagram of Feedback Subsystem

The screen communicated with the control subsystem via SPI protocol. A TFT library designed to be compatible with the Arduino integrated development environment (IDE) and the ESP32, TFT_eSPI, was used to write text to the display. This library was chosen for its vast configurations, ease of use, and availability on the Arduino IDE platform. Using this library, the background color, font color, font size, and screen orientation were set. When attempting to communicate with the screen with VSPI, the screen would faintly flash white and appeared to be resetting continuously. Upon looking into this issue, it appears as though there is an issue in some versions of Arduino IDE that gets stuck in the setup loop when trying to use VSPI. A solution to this problem was to change to the HSPI general-purpose SPI peripheral.

To light the LED arrays, a weighted average of student responses was taken and translated to an integer between 1 and 15 to indicate the number of LED sets that should be lit. In the application, students have the option to rate their understanding from a scale of 1 to 5. Referring to the chosen understanding level as “n”, the weighted average used a sum of “n-1” values in the numerator and the number of responses multiplied by four in the denominator. This allowed a full range of 0 to 1 to be utilized to light the LEDs, with 0 indicating no understanding and 1 indicating full understanding. The weighted average was multiplied by 15 and rounded up to determine how many sets of LEDs should be lit.

2.4 Mobile App Subsystem

The function of the mobile app system is to serve as the user interface that allows users to send data, adjust settings, and overall interact with the central hub. Although a web app would have had the same functionality, the decision to go with a mobile app considered that phones take up less space on the desk and most students will always have their phone, but not necessarily their laptop. The app was created in Android Studio using Flutter because it offered cross-platform development, unlike the other common platform, Xcode, that only develops for iOS. Flutter also offers a variety of widgets such as buttons, switches, sliders, and text fields that would support the app functionality well. It also supports performance on a standard smartphone as it operates at 60 or 120 FPS depending on the refresh rate of the device.

To avoid connecting to the hub's Bluetooth via the mobile device's settings, the flutter_blue_plus dart package allowed the device to scan for nearby BLE devices and connect directly from the app [8]. This package prepares the data packets for transmitting and receiving. Flutter_blue_plus was the natural choice for this project because the team has used it in past projects, and it is also up to date with the current Android Studio version.

When designing the app layout, the priority was to have an intuitive user interface (UI). This means, for example, using large buttons for the understanding rating selection, adding labels to any widgets the user could interact with, and contributing spacing between the widgets so there are no accidental presses. Images illustrating the app flow and page layouts are detailed in Table 12 (Appendix D). To ensure that the page layouts would work across various phones, the app was tested on the Android Studio emulator and a physical Pixel 4 phone.

Within the app there are five possible screens the user can encounter: A welcome page, a student login page, an instructor login page, a student homepage, and the hub settings. The welcome page allows users to select their role as either student or instructor, which leads to their respective login process. Unlike instructors, students are required to input a display name. After that, both the students and instructors follow the same process of connecting to the BLE by selecting the "Hub_1" device name and inputting their respective passwords.

Once logged in, the user is brought to the homepage. Students are brought to a homepage intended for live interaction. This includes button options to submit a 1-5 level of understanding, a "raise hand" button for live questions, and a text field for typed out questions with an option to toggle anonymous submission. On the other hand, instructors are brought to the hub settings page which contains a slider to change the font size and a ON/OFF toggle for the vibration motor. A flowchart detailing the functionality of the app can be found in Figure 19 (Appendix C).

3. Design Verification

3.1 Power Management Subsystem

Entries 1-3 in Table 11 (Appendix A) describe the requirements and verifications of the proper function of the power management subsystem. Requirement 1 determines the acceptable range of operation for the power adapter to be at least 1.5 A at 5 V +/- 0.5 V. The current was confirmed by the proper function of all required components; with insufficient current this would not have been the case. The 0.5 V tolerance was selected by examining the datasheets of the hub components and determining the maximum and minimum voltage they would be able to function with. Entries 1-3 of Table 3 confirm that the adapter requirements were met with 5.056 V being well within the 5 V +/- 0.5 V specification. Skipping to Requirement 3, it similarly determines the acceptable operation of the DC/DC converter module to be at 3.3 V +/- 0.3 V. The last two entries of Table 3 confirm that these requirements were also met with the test point measurement of 3.291 V being within 0.3 V of 3.3 V.

Table 3: Multimeter measurements of power management subsystem test points

Test Point	Voltage [V]
Power Adapter (unconnected)	5.059 V
Barrel Jack with power adapter plugged in	5.056 V
5 V Test Point (Vin)	5.056 V
3V3 Test Point (Vout)	3.291 V
3V3 Test Point (Motor)	3.290 V

Returning to Requirement 2, it defines the functional operation of the power adapter to be safe and stable when the temperature of the device is below 45°C, the threshold of safe human touch [12]. There also are no significant variations in the input voltage over the period of one lecture (about 50 minutes). Table 4 verifies that at no point during a lecture period, does the hub or any of its accessible components approach an unsafe temperature threshold. Figure 9 shows the constant voltage signal from the oscilloscope with no discernable spikes that would cause disruption.

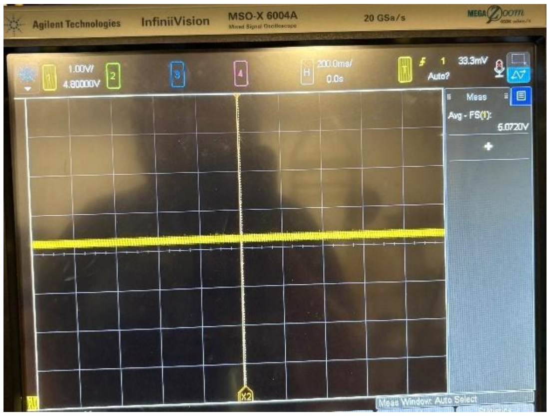


Figure 9: Oscilloscope measurement of adapter voltage input during 50-minute testing period

Table 4: Temperature measurements of significant hub surfaces taken after 50 minutes of being plugged in

Component	Temperature [°C]
Block	24.9
Barrel Jack	33.5
Casing	30.3

3.2 Control Subsystem

Requirements 4-7 in Table 11 pertain to the control subsystem (Appendix A). Requirement 7 sets the accepted high and low input voltage ranges for the button GPIOs. Similarly, Requirement 8 sets the accepted high and low input voltage ranges for the rotary encoder. These were determined by the ESP32-S3 datasheet [1]. The results of the tests verify the voltages fall in the allowable ranges and are listed in Table 5.

Table 5: Measured Bluetooth connection time over multiple trials

Component	High GPIO [V]	Low GPIO [mV]
Clear Button	3.285	28
Rotary Encoder A Signal	3.264	-80
Rotary Encoder B Signal	3.263	-110

Requirement 6 outlines the forward voltage needed by the LED arrays and voltage to remain off. Likewise, Requirement 7 states the voltage needed by the BJT to turn on the vibration motor and the voltage to remain off. These voltages are produced as outputs from the ESP32 GPIOs. One example of each array and the vibration motor measured voltages are shown in Table 6.

Table 6: Measured Bluetooth connection time over multiple trials

Component	High GPIO [V]	Low GPIO [mV]
Red Array Set 1	1.95	0
Yellow Array Set 2	2.113	0.14
Green Array Set 3	2.118	0.09
Vibration Motor	2.5	0.185

3.3 Feedback Subsystem

Requirements 8-11 in Table 11 pertain to the feedback subsystem (Appendix A). Requirements 8 and 11 provide the maximum time the LEDs and the vibration motor have to respond to incoming data from the application. This time was based on avoiding the progression of class material before the questions or engagement level could be viewed. These times were not validated using the given procedures as a micro-USB port was not attached to the PCB to use the data signals provided by the ESP32. However, upon visual inspection, the reaction time of all components was nearly instant as the app was interacted with.

Requirement 9 ensured that the rotary encoder must react correctly when being turned. This is determined by verifying signal A leads signal B during clockwise rotation and signal B leads signal A during counterclockwise rotation. The oscilloscope captures in Figure 10 validates this behavior.

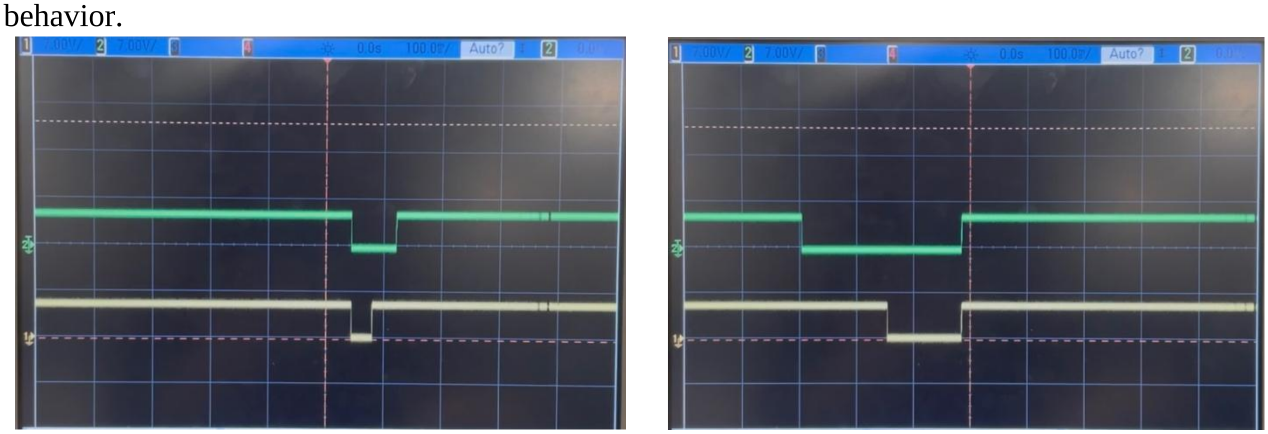


Figure 10: Signal A (yellow) and signal B (green) for clockwise (left) and counterclockwise rotation (right)

Requirement 10 established a character count that must display on the screen with the largest font size. A question was sent to the hub after reaching the character limit of 200 words on the application. As seen in Figure 11, the screen successfully displayed the maximum character count.

Figure 11: Hub screen displaying 200 characters

3.4 Mobile App Subsystem

Requirements 12-15 in Table 11 pertain to the mobile app subsystem (Appendix A). Requirement 12 addresses the delay between when the user presses the “Connect” button and when the Bluetooth connection is made. Table 7 lists a series of trials conducted using the verification method. The results showed an average connection time of 1.5238304 seconds, which is under the 10 second requirement.

Table 7: Measured Bluetooth connection time over multiple trials

Trial	**Start Time** HH:MM:SS	**End Time** HH:MM:SS	**Connection Time** seconds
1	11:27:55.106995	11:27:56.639120	1.532125
2	21:59:52.549592	21:59:54.077163	1.527571
3	22:00:49.428784	22:00:50.637720	1.208936
4	22:02:03.954799	22:02:05.220084	1.265285
5	22:02:53.939996	22:02:56.025231	2.085235

Requirement 13 investigates the discrepancies between the data sent from the app and the data received by ESP32. As shown in Table 8, the data transmission had 100% accuracy when tested with all the available methods of data submission in the app.

Table 8: Data transmission accuracy from app to ESP32 using different submission methods

Trial	Submission Method	Send Data	Received Data	Error (%)
1	Raise Hand Button	“K/Has a LIVE question%”	“K/Has a LIVE question%”	0
2	Text Submission	“K/Test 123%”	“K/Test 123%”	0
3	Text Submission	“Anonymous/Hello World%”	“Anonymous/Hello World%”	0
4	Engagement Button	“5/3%”	“5/3%”	0
5	Font Slider	“3.0/false%”	“3.0/false%”	0
6	Silent Mode Switch	“3.0/true%”	“3.0/true%”	0

Requirement 14, which specifies that the app should respond to any user interactions within one second, was not associated with any quantitative data. However, the requirement was verified qualitatively as the app reacted instantly to user interactions during final functionality testing.

As mentioned in section 2.4, the app was created using Android Studio. Although Android Studio allows for dual design for both Android and iOS development, uploading the app to an iOS device requires Xcode which is only offered on Mac. Since there was not a Mac readily available and loaded with Xcode, only the one Android phone on hand could be loaded with the app and therefore Requirement 15 could not be tested.

4. Costs and Timeline

4.1 Parts and Labor

Table 9 estimates the cost for producing the ClassroomClarity Hub. The average salary of a UIUC electrical engineering graduate is $88,000 per year, which is equivalent to $42 hourly pay [9]. Equation 5 calculates the cost of labor for each team member [10].

$$Labor\ Cost = \ ideal\ salary\ (hourly\ rate) \times hours\ worked \times 2.5 \qquad (5)$$

Estimating that each team member worked six hours per week for 14 weeks, the total cost of labor is $26,460. Therefore, the total cost of production is $\$26{,}460 + \$78.26 = \$26{,}538.26$.

Table 9: Cost Analysis for ClassroomClarity Production

Labor					
Team Member	**$/hour**	**Hours Worked/Week**	**Weeks Worked**	**Base Cost ($)**	**X2.5 Cost ($)**
Maddie Donku	$42	6	14	3,528	8,820
Kaitlin Gowens	$42	6	14	3,528	8,820
Jesse Gruber	$42	6	14	3,528	8,820
				Total:	$26,460
Parts					
Description	**Part Number**	**Manufacturer**	**Quantity**	**Unit Cost ($)**	**Total Cost ($)**
USB-C to DC Power Cord	B0D4F42GW2	FASTELECTRIC	1	6.49	6.49
Barrel Jack	694106301002	Würth Elektronik	1	0.92	0.92
DC/DC Converter	TPSM84203EAB	Texas Instruments	1	5.16	5.16
0.01uF Capacitor	C0805C103K5RACTU	KEMET	2	0.30	0.60
0.1uF Capacitor	CL21F104ZAANNNC	Samsung Electro-Mechanics	1	0.00*	0.00
1uF Capacitor	CL21B105KBFNNNG	Samsung Electro-Mechanics	3	0.00*	0.00
10uF Capacitor	GRM21BR61H106ME43L	Murata Electronics	1	0	0.00
270Ω Resistor	RC0805JR-07270RL	YAGEO	1	0.01	0.01
10kΩ Resistor	RMCF0805JG10K0	Stackpole Electronics	7	0.00*	0.00
Protection Diode	1N4002	Onsemi	1	0.00*	0.00
BJT	BD139	STMicroelectronics	1	0.67	0.67

Rotary Encoder	PEC16-4120F-N0012-ND	Bourns Inc.	1	3.38	3.38
Push Button	1568-14460-ND	Sparkfun Electronics	1	2.01	2.01
Red LED Array	DC10EWA	Kingbright	1	0.00*	0.00
Yellow LED Array	DC10YWA	Kingbright	1	0.00*	0.00
Green LED Array	LTA-1000G	Lite-On Inc	1	0.00*	0.00
Blue LED	151051BS04000	Würth Elektronik	1	0.00*	0.00
LCD Screen	MSP4002	Hosyond	1	18.99	18.99
Vibration Motor	316040004	Seed Technology Co., Ltd	1	3.60	3.6
Push Button	PTS645SL43-2 LFS	C&K	2	0.00*	0.00
2 Wire Screw Header	732-691210910002-ND	Würth Elektronik	2	10.30	20.6
3 Wire Screw Header	732-691210910003-ND	Würth Elektronik	1	1.39	1.39
4 Pos Header	61300411121	Würth Elektronik	2	0.00*	0.00
ESP32 Microcontroller	ESP32-S3-WROOM-1-N16	ESPRESSIF	1	0.00*	0.00
Programmer	SH-U09C	DSD TECH	1	12.49	12.49
Jumper Wire M To F	1528-1964-ND	Adafruit Industries LLC	1	1.95	1.95
Test Point	5012	Keystone Electronics	11	0.00*	0.00
*Received from the Electronic Services Shop					
				Total	78.26

4.2 Schedule

Table 10 outlines the timeline followed by the ClassroomClarity team to stay on task and complete the hub over the course of one semester.

Table 10: ClassroomClarity project timeline

Week of	Task	Group Member(s)
2/17	Divide tasks	All
	App and Bluetooth research	Kaitlin
	Control signals and programmer research	Jesse
	Power management research	Maddie
2/24	Component selection and ordering	All

	1st Draft of PCB modeling on KiCad	Maddie
	App Design	Kaitlin
	Control signal organization and planning	Jesse
	Order 1st PCB	Maddie
	Document power management subsystem, budget, timeline, and risks	Maddie
	Document control and feedback subsystems and tolerance analysis	Jesse
	Document user interface subsystem and software design	Kaitlin
	Prepare for breadboard demo	All
	Get basic student side components functional (Understanding rating buttons, text submission, raise hand button)	Kaitlin
3/3	Begin CAD modeling for enclosure	Kaitlin
	Breadboard Demo	All
	Finish the microcontroller code for controlling clear button, notification LED, and LCD screen	Kaitlin
	Finalize GPIO pin configurations	Jesse
3/10	Get all feedback systems working with test data set from microcontroller	All
	Assemble PCB	Maddie
	Test PCB & Discuss improvements	All
	Work on hub enclosure	Jesse
	Edit and order 2nd PCB	Maddie
3/24	Start Bluetooth code development for sending data from app to ESP32	Kaitlin
	Get microcontroller software working so all feedback systems react properly to incoming data	Jesse
	Finish Bluetooth code development for sending data from app to ESP32	Kaitlin
	PCB Debugging	Maddie
3/31	Discuss prototype improvements	All
	Power Debugging	Maddie
	Add login page with functional student and teacher login + password	Kaitlin
4/7	Edit and print final casing design	Jesse
	PCB Version 2 Assembly	Maddie
	Version 2 PCB testing	All
4/14	Merge the functional Bluetooth and Microcontroller Code	Kaitlin
4/21	Mock Demo	All

	Assemble Final Product	All
	Final Demo	All
	Mock Presentation	All
4/28	Finalize Presentation	All
	Final Presentation	All
5/5	Complete Final Report	All

5. Conclusion

5.1 Accomplishments

Over the course of 14 weeks, the ClassroomClarity Hub went from an idea to a polished product. The high-level goals set at the beginning of its journey were accomplished. A student was able to send questions to the hub so that a professor could view the question on a screen. Up to five questions could be held on the hub with one question displayed at a time. A professor could scroll through and clear these questions using the featured button and dial. The hub successfully alerted the central user quickly after a question was asked through vibrating the hub and lighting an indicator light. Additionally, LED arrays were quickly manipulated upon receiving a new level of understanding from a user.

Apart from reaching the main goals, the hub was able to be connected to a portable power source if desired. A housing was created that could eject rising heat from the electronics inside. Aside from what was accomplished, the opportunities for further development and refinement brought curiosity and excitement about what could be next. Also, the team was able to exercise and further develop their technical skills with a product that could be distributed outside of the lab. Overall, the process of realizing, designing, and assembling a useable product was very enjoyable and concluded in pride.

5.2 Uncertainties

During the design of the ClassroomClarity Hub, the team faced a few challenges. One of the major challenges was an issue where the wall to PCB adapter was not delivering the proper power. To remedy this issue, several steps were taken to diagnose and debug. First, a multimeter was used to check for shorts on the board, one was found and repaired but this was not the main cause of the power discrepancy. Next, a different source of power was tried, a DC supply from the 445 Lab, and it was verified that the circuit itself functioned properly. With this verification, it was able to be determined that the fault lay in the power adapter itself. Upon closer inspection of the data sheet and comparison to other available models, the adapter being used had a reversed polarity from the standard. A new adapter of the correct polarity was ordered and the system worked as intended.

5.3 Ethical Considerations

Throughout the development of the ClassroomClarity Hub, the ethical impacts of the product were kept in mind. First, it is important to treat all people fairly and respectfully, which includes not discriminating against, harassing, or injuring others [11]. The hub would receive questions from many people over multiple lectures in a day. Therefore, it must be ensured that each person has equal opportunity to use the hub. Since the data of consumers is being handled, the product must be responsible with this data, ensuring the privacy of consumers in maintained. Lastly, the product should be manufactured in a sustainable way by prioritizing material selection and reducing waste associated with the creation of the product.

5.4 Future work

Due to the time constraint of producing a product in a single semester, there are some additional features that the team would have liked to add given more time. In terms of continued testing of the ClassroomClarity Hub product, connecting multiple mobile devices would be beneficial. Certain roadblocks in the operating system of Apple iPhones prevented the download of the ClassroomClarity app on such devices. Since the team only had one Android phone, Bluetooth testing could only be completed with that one device. In theory, multiple devices should be able to connect to the hub functionally but verifying this fact would be ideal. As for additional software features, an in-app question queue and a hub question counter were discussed. These features add quality of life improvements to the ClassroomClarity experience by allowing students to view upcoming questions from their peers, as to not repeat similar questions and for the professor to view how many questions are waiting to be answered. The in-app queue would require additional moderation to adhere to proper ethics and ensure that viewable questions remain professional. For physical improvements to the hub, a rotary encoder with a smaller number of pulses would be selected to make flipping through questions smoother and easier. The final improvement discussed would be to add a stand to the hub to be able to adjust the angle of the casing for the professor to better view the screen. Overall, the ClassroomClarity team was very satisfied with the final product and additional features would only enhance the hub.

References

[1] Espressif Systems, *ESP32-S3-WROOM-1 ESP32-S3-WROOM-1U Datasheet* (2021). Accessed: May 6, 2025.[Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32 -s3-wroom-1_wroom-1u_datasheet_en.pdf

[2] LCD WIKI. *4.0inch SPI Module MSP4022&MSP4023 User Manual*. (2021). Accessed: May 7, 2025. [Online]. Available: http://www.lcdwiki.com/res/MSP4022/4.0inch_SPI_Module_MSP4022&MSP4023_User_Manual_EN.pdf

[3] Seeed Technology Co., Ltd. *Product Specification*. (2016). Accessed: May 7, 2025. [Online]. Available: https://mm.digikey.com/Volume0/opasdata/d220001/medias/docus/2226/1027_Spec.pdf

[4] Kingbright. *WP5603QBDL/SD/G Oval Solid State Lamp*. (2018). Accessed: May 7, 2025. [Online]. Available: https://www.kingbrightusa.com/images/catalog/SPEC/WP5603QBDL -SD-G.pdf

[5] Texas Instruments. *TPSM84203, TPSM84205, TPSM84212 1.5-A, 28-V Input, TO-220 Power Module*. (2017). Accessed: May 7, 2025. [Online]. Available: https://www.ti.com/lit/ds/symlink/tpsm84203.pdf?HQS=dis-dk-null-digikeymode-dsf-pf-null-wwe&ts=1741317560705&ref_url=https%253A%252F%252Fwww.ti.com%252Fgeneral%252Fdocs%252Fsuppproductinfo.tsp%253FdistId%253D10%2526gotoUrl%\253Dhttps%253A%252F%252Fwww.ti.com%252Flit%252Fgpn%252Ftpsm84203

[6] J. Christofferson. “Switch Bounce and How to Deal with It,” *All About Circuits*. (2015), Accessed: 03/31/2025. [Online]. Available: https://www.allaboutcircuits.com/technical-articles/switch-bounce-how-to-deal-with-it/#:~:text=%22When%20the%20switch%20is%20closed,any%20sort%20of%20bounce%20control:.

[7] Arduino, *ArduinoBLE*. Arduino.cc. [Online]. Available: https://docs.arduino.cc/libraries/arduinoble/#BLEDevice%20Class

[8] chipweinberger, *flutter_blue_plus*, GitHub. [Online]. Available: https://github.com/chipweinberger/flutter_blue_plus/blob/master/packages/flutter_blue_plus/example/lib/main.dart [Accessed: May 6, 2025]

[9] The Grainger College of Engineering, 2023, “Post-Graduation Success,” The Grainger College of Engineering [Online]. Available: https://grainger.illinois.edu/academics/undergraduate/majors-and-minors/electrical-engineering.

[10] The staff of ECE 445 and ECE Editorial Services, “PREPARING YOUR FINAL REPORT FOR ECE 445, SENIOR DESIGN,” Available: https://courses.grainger.illinois.edu/ece445/documents/ECE_445_Final_Report_Guidelines.pdf

[11] IEEE. “IEEE Code of Ethics.” (2020), [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html (visited on 02/11/2025).

[12] E. Ungar and K. Stroud. “A New Approach to Defining Human Touch Temperature Standards,” NASA/Johnson Space Center, Houston, TX, USA. Accessed: May 7, 2025. [Online]. Available: https://ntrs.nasa.gov/api/citations/20100020960/downloads/20100020960.pdf

Appendix A Requirement and Verification Table

Table 11: System Requirements and Verifications

Subsystem	Requirement	Verification	Verification status (Y or N)
Power Management	1. The AC DC wall adapter must supply at least 1.5A to the hub and 5V +/- 0.5V to the DC/DC converter module.	1. Ensure adequate connection of adapter to barrel jack. 2. Probe 5V and ground test points which correspond to adapter output with multimeter. 3. Confirm that measured voltage is within 4.5-5.5V	Y
	2. The adapter must be able to safely provide a stable supply of power at the very least the length of one lecture (50 minutes).	1. Plug in clarity hub to wall adapter and ensure adequate connection. 2. Probe 5V and ground test points which correspond to adapter output with oscilloscope. 3. Observe voltage and current signals from oscilloscope for the duration of a lecture period. 4. Use an infrared thermometer to measure adapter temperature at both the wall outlet and barrel jack connections and ensure the temperature is below 45°C (limit for safe human touch [12]) 5. Confirm that there are no voltage or current discrepancies during testing period	Y
	3. The DC/DC converter module must take the input from the AC DC adapter and output 3.3V +/- 0.3V.	1. Ensure adequate connection of adapter to barrel jack. 2. Probe 3V3 and ground test points corresponding to converter output with multimeter. 3. Confirm that measured voltage is within 3.0-3.6V	Y
Control	4. The unpressed input voltage of each button GPIO must fall between 2.5-3.6V, and the	1. Probe test points of each button using a multimeter. 2. Confirm the “unpressed”	Y

	pressed input voltage must be between -0.3-0.8V within one second of pressing the button.	voltage is between 2.5-3.6V 3. Confirm the “pressed” voltage is between -0.3-0.8V within a second of pressing the button	
	5. The rotary encoder processed signals must have a maximum between 2.5-3.6V and a minimum between -0.3-0.8V.	1. Probe test points of processed signals RotA and RotB using an oscilloscope. 2. Begin turning the encoder until voltage is high on RotA and RotB 3. Confirm the maximum voltage is between 2.5-3.6V and the minimum voltage is between -0.3-0.8V	Y
	6. The high output voltage of the green, yellow, and red LED GPIO pins must be greater than 1.9V and the low output voltage must be less than 1.5V.	1. Probe the headers of the LEDs using a multimeter. 2. In the microcontroller code, set the LED GPIO pins high. 3. Upload the code to the microcontroller. 4. Confirm the header voltage is greater than 1.9V 5. In the microcontroller code, set the LED GPIO pins low. 6. Upload the code to the microcontroller. 7. Confirm the header voltage is less than 1.5V	Y
	7. The high output voltage of the vibration motor GPIO pin must be greater than 2V and the low output voltage must be less than 1V.	1. Probe the test point of the vibration motor control signal. 2. In the microcontroller code, set the vibration motor GPIO pin high. 3. Upload the code to the microcontroller. 4. Confirm the voltage is greater than 2V. 5. In the microcontroller code, set the vibration motor GPIO pin low. 6. Upload the code to the microcontroller. 7. Confirm the voltage is less than 1V in the microcontroller code	Y
Feedback	8. LEDs should react to any changes in “Understanding	1. In the microcontroller code, add a “t_start” variable right	Y

	Rating" data sent by the app within 30 seconds.	after the get data function. Add a "t_end" variable after the change to the "Understanding Rating" LEDs GPIO signal. Both variables should store the time at occurrence. 2. At the end of the code, add a line to print out t_end - t_start to the serial display. 3. Upload the code to the microcontroller and restart the device and reconnect the app. 4. Select a new understanding rating button. The hub's LED should update to reflect the new value within 30 seconds.	
	9. Signal A of the rotary encoder must lead signal B when turning the encoder clockwise. Signal B must lead signal A when turning counterclockwise.	1. Probe test points of processed signals RotA and RotB using an oscilloscope triggered on a rising edge. 2. Turn the encoder clockwise and confirm signal A's waveform leads signal B's. 3. Turn the encoder counterclockwise and confirm signal B's waveform leads signal B's	Y
	10. The LCD should display a maximum of 200 characters with a black background and white font	1. In the microcontroller code, utilize the TFT_eSPI library included in the Arduino IDE to run a provided demo. 2. Upload the code to the microcontroller. 3. Confirm the demo's operation on the screen	Y
	11. The vibration motor should vibrate within 30 seconds of a question being sent from the app	1. In the microcontroller code, add a "t_start" variable right after the get data function. Add a "t_end" variable after the change to the vibration motor GPIO signal. Both of these variables should store the time at occurrence. 2. At the end of the code, add a	Y

		line to print out t_end - t_start to the serial display. 3. Upload the code to the microcontroller and restart the device and reconnect the app. 4. Send a question to the system. The vibration motor must begin vibrating within 30 seconds.	
Mobile Application	12. The app must Bluetooth connect to ESP32 within 10 seconds of initiating a connection request	1. Within the app code, add a print statement containing the current time that activates when the “connect” button is pressed. 2. Also in the app code, add another print statement that prints the current time when the connection process completes. 3. Upload the code to the app. 4. On the app, try to connect to the microcontroller. 5. Once it finishes connecting, read the terminal and confirm the time between start and finish is <10s.	Y
	13. The app should be able to send and receive data with a less than 1% error rate per transmission.	1. In the microcontroller code, add print statements that print out the received data. 2. Upload the new code to the microcontroller and Bluetooth connect the app to the microcontroller. 3. On the app, select a different “Understanding Rating” button. 4. Check serial display to confirm the received understanding rating matches. 5. On the app, press the “Raise Hand” button. 6. Check serial display to confirm the received display name matches. 7. On the app, submit a text question. 8. Check serial display to confirm the received question matches	Y

	14. The app should respond to any user interactions within 1 second.	1. In the app, select a new "Understanding Rating" button. 2. Confirm the app reflects the change within 1 second of pressing the button.	Y
	15. The app should hold any data ready to be transmitted until the central hub's signals it is ready to accept new data.	1. Connect 2 devices to the hub. 2. Send in two different text questions at the same time. 3. Use the rotary encoder to scroll through the questions queue and confirm both questions were received.	N

Appendix B LTSpice Simulation

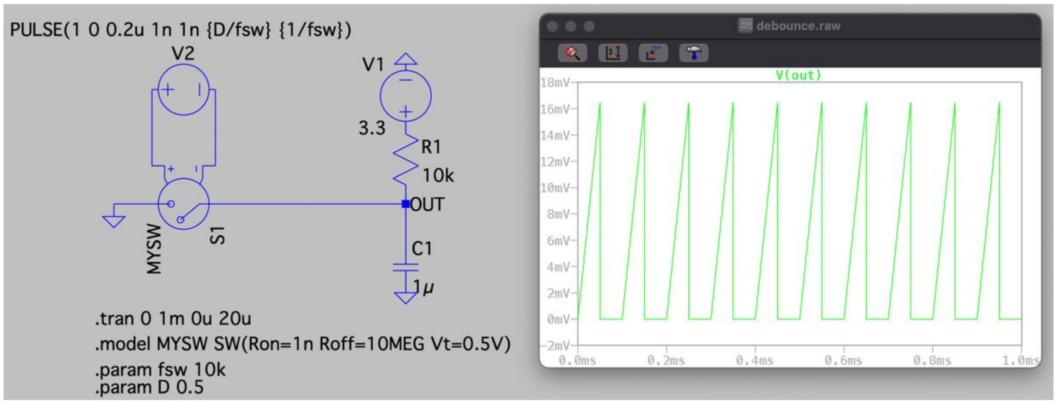


Figure 12: LTSpice simulation of button RC circuits during bounce

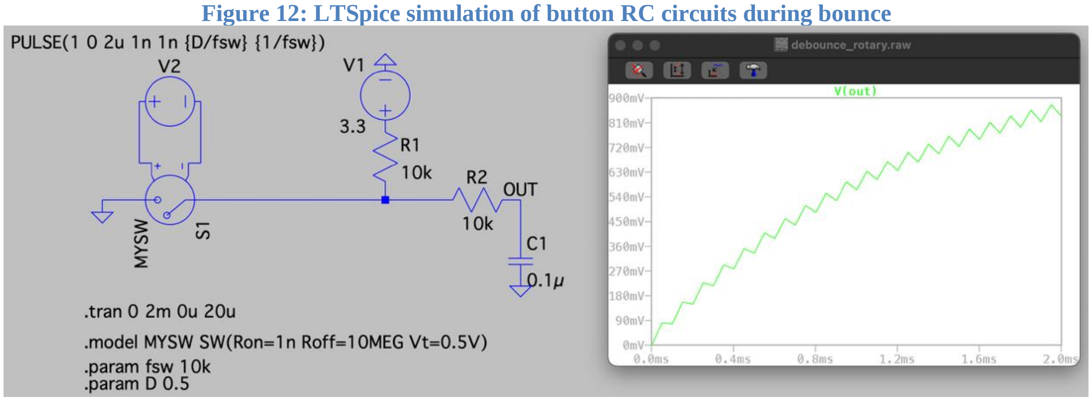


Figure 13: LTSpice simulation of rotary RC circuit during bounce

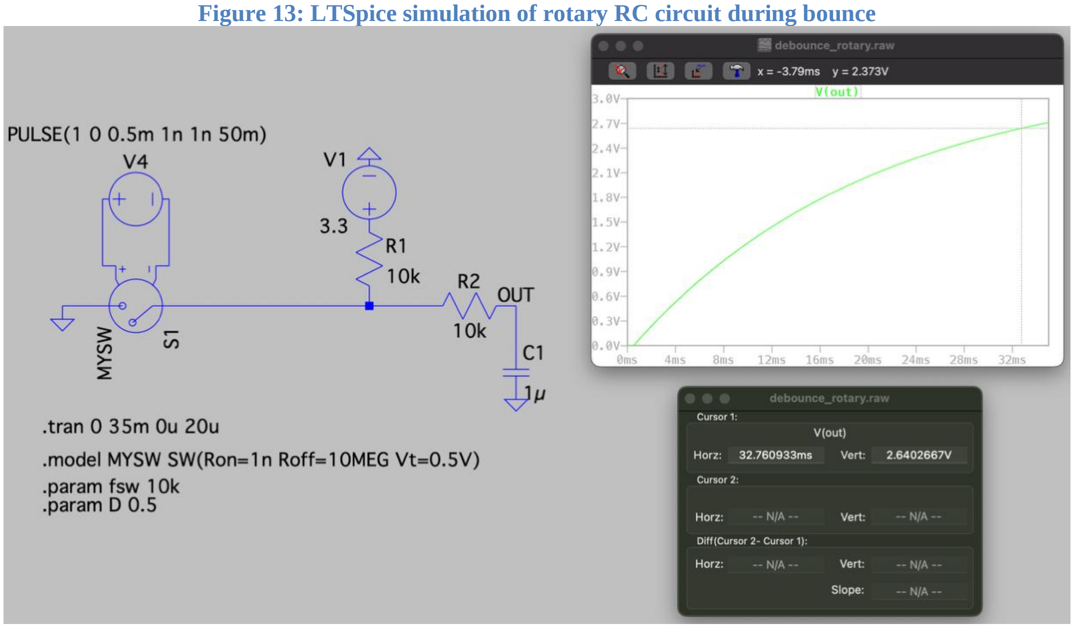


Figure 14: LTSpice simulation of rotary RC 1 µF circuit from 0 V to 2.64 V

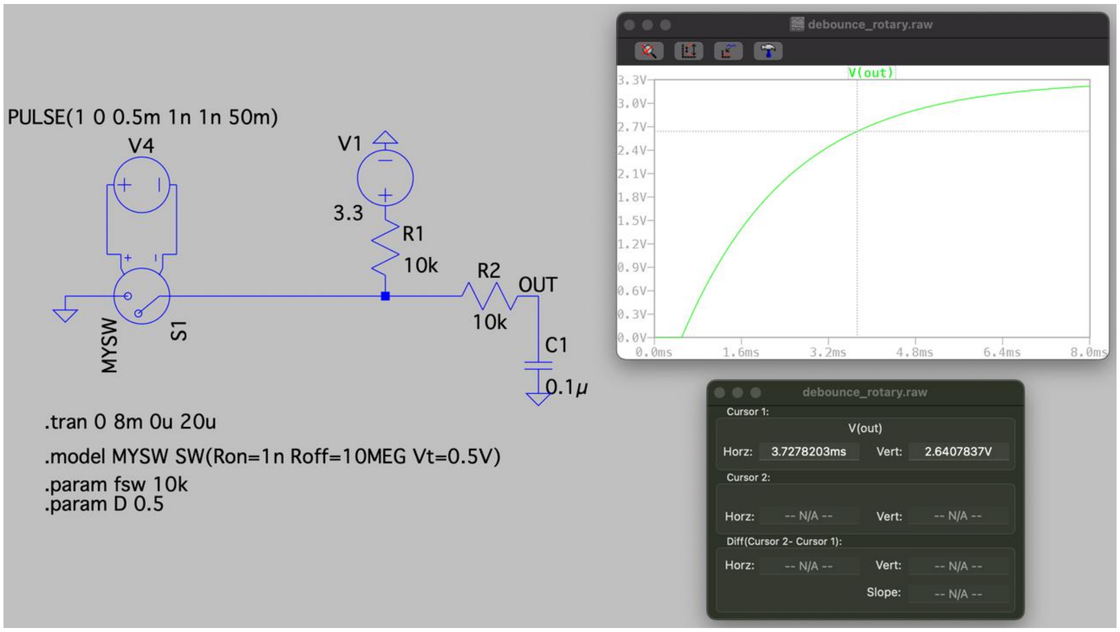


Figure 15: LTSpice simulation of rotary RC 0.1 µF circuit from 0 V to 2.64 V

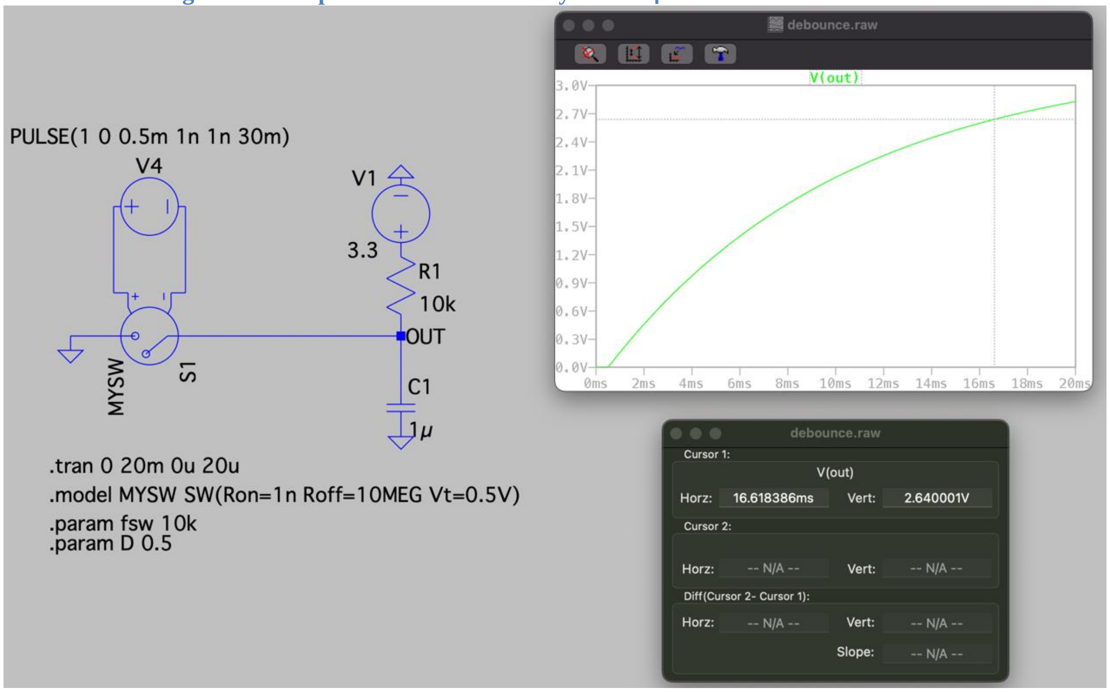


Figure 16: LTSpice simulation of button RC 1 µF circuit from 0 V to 2.64 V

Appendix C Software Flowcharts

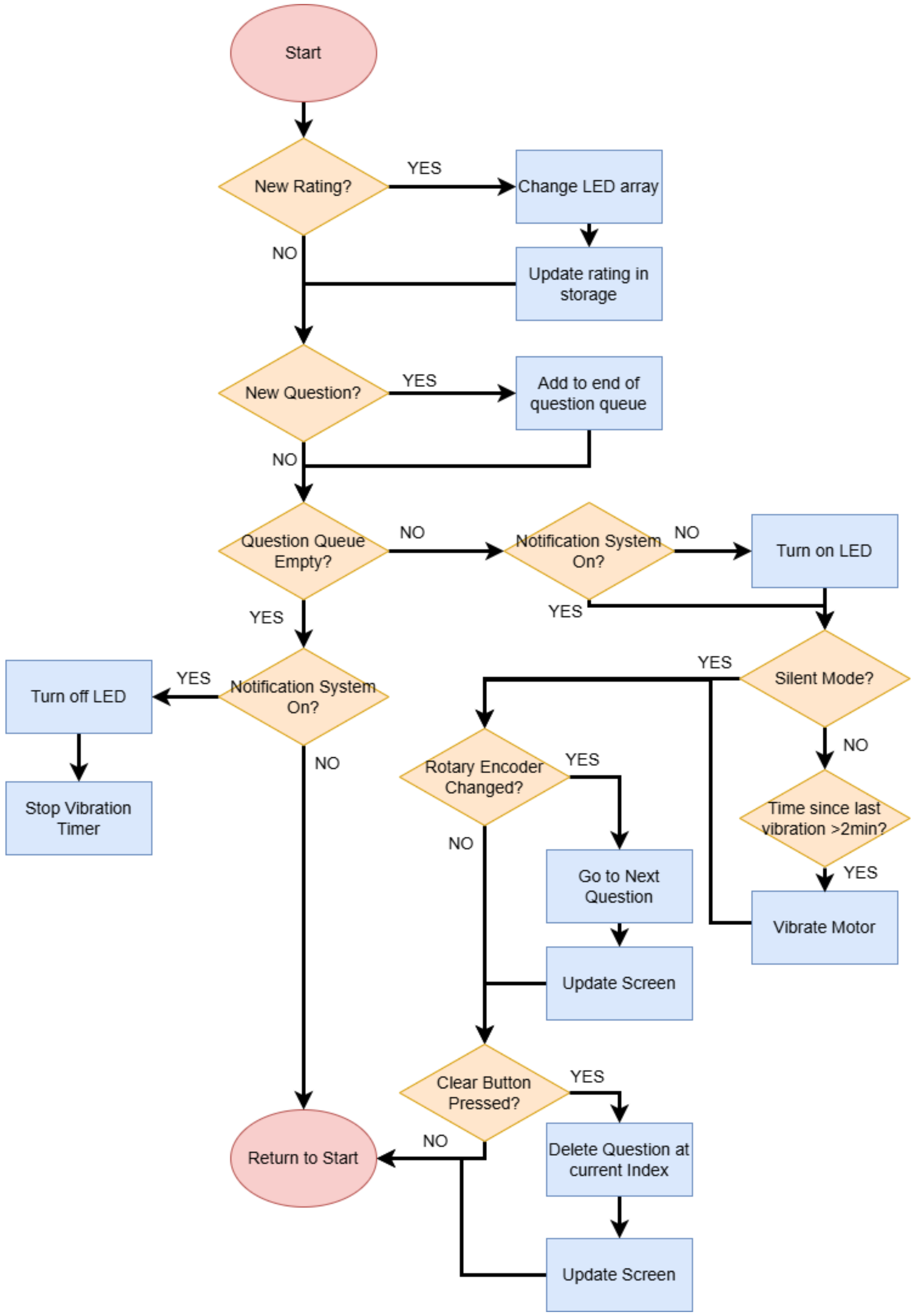


Figure 17: Flowchart detailing how the ESP32 handles input and output signals

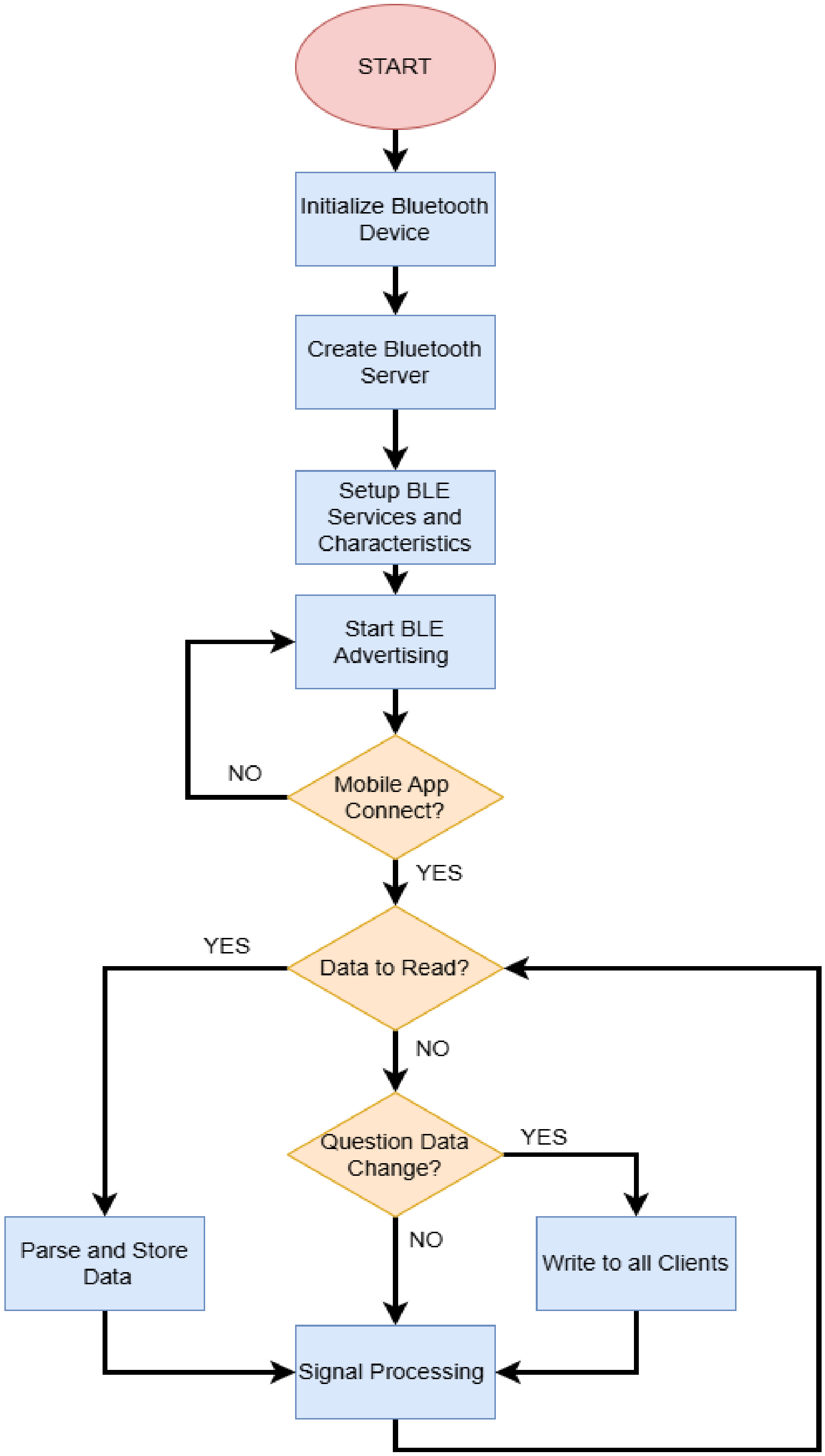


Figure 18: Flowchart of the ESP32 BLE process from initialization to operation

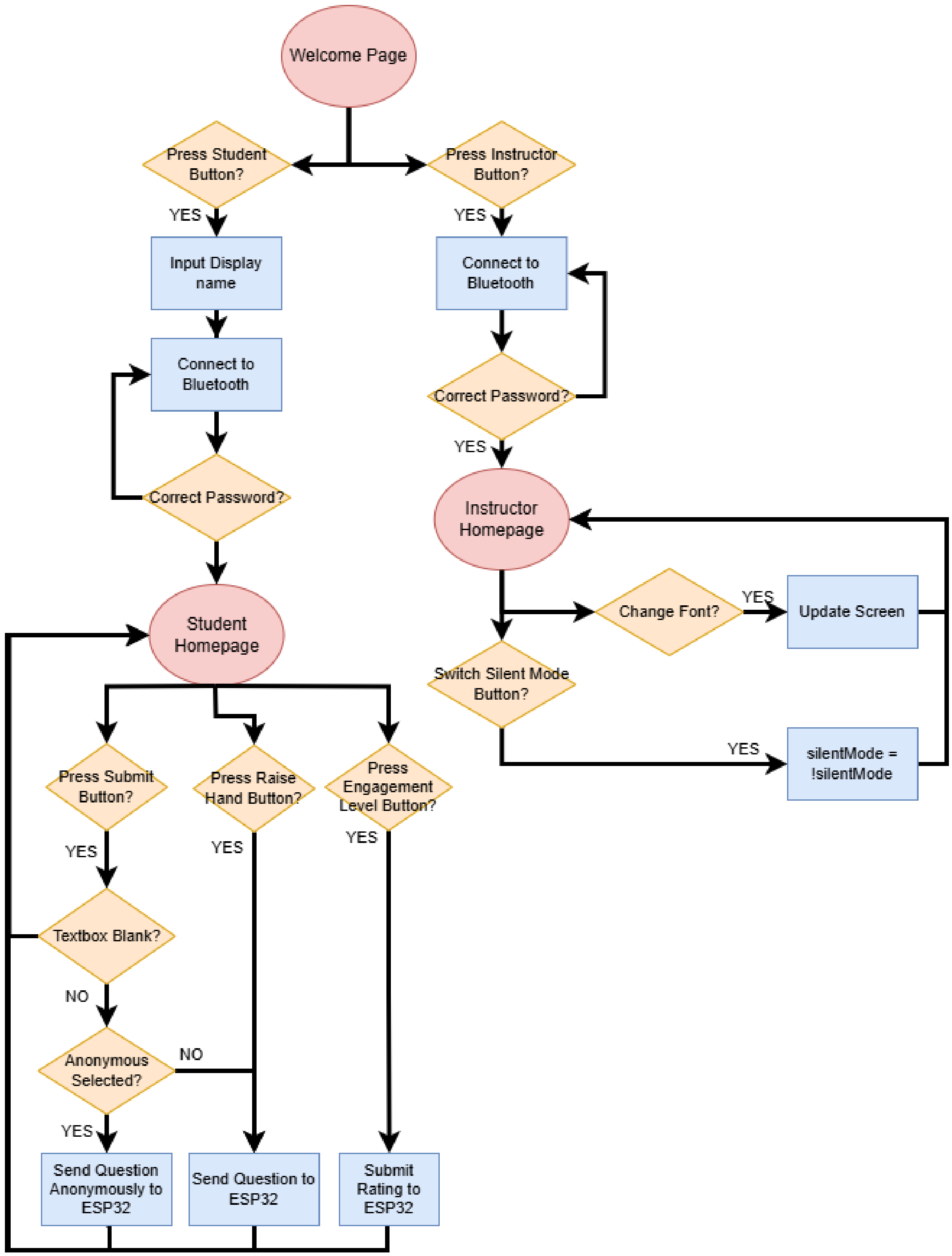
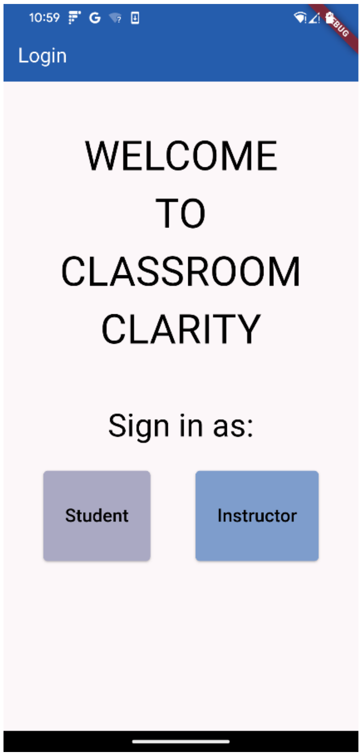


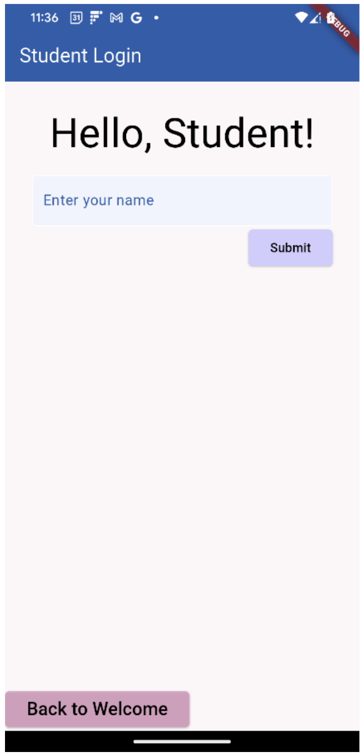
Figure 19: Flowchart for the mobile app functionality

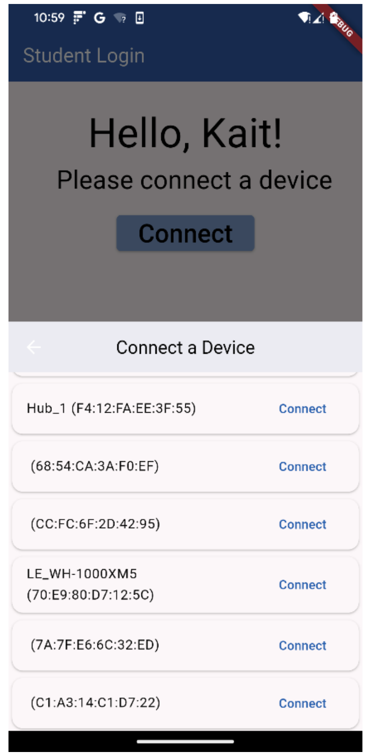
Appendix D Mobile App Screen Layouts and Logic

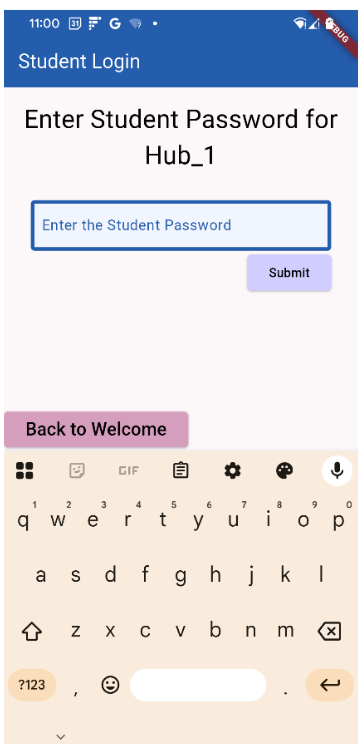
Table 12: The flow of mobile app following a student and instructor user. Read from left to right, top to bottom.

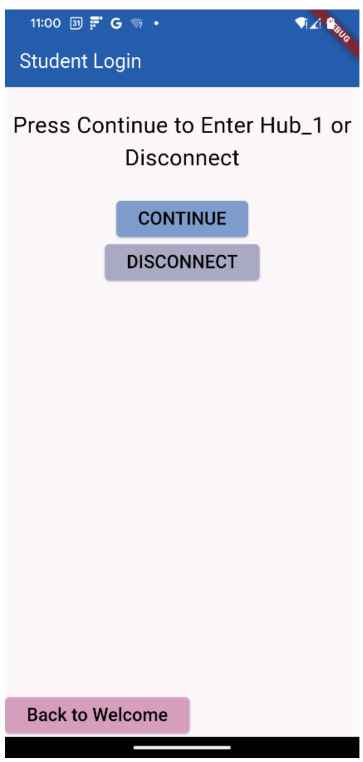
Student Path

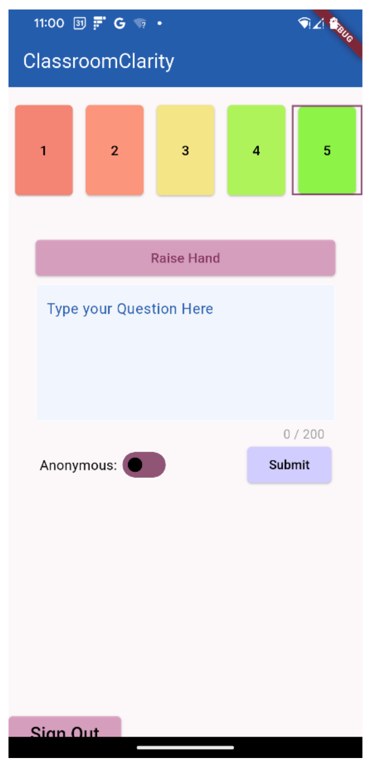




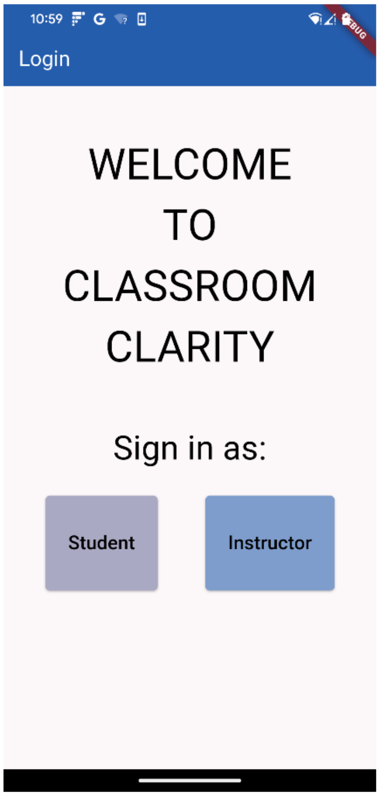


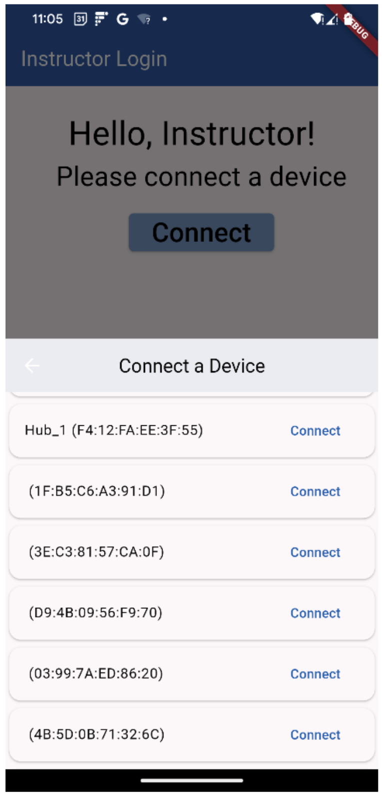


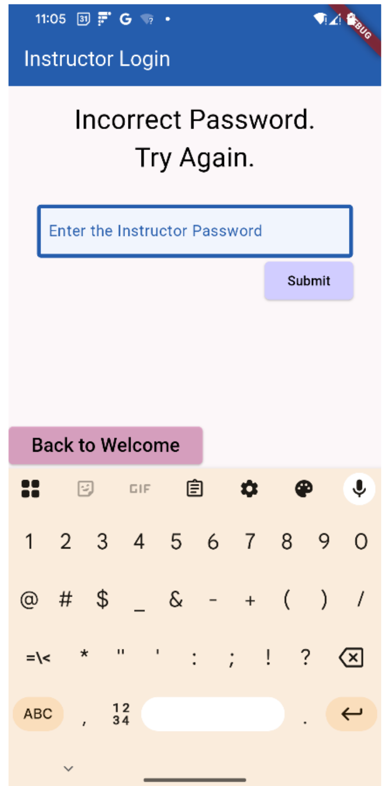


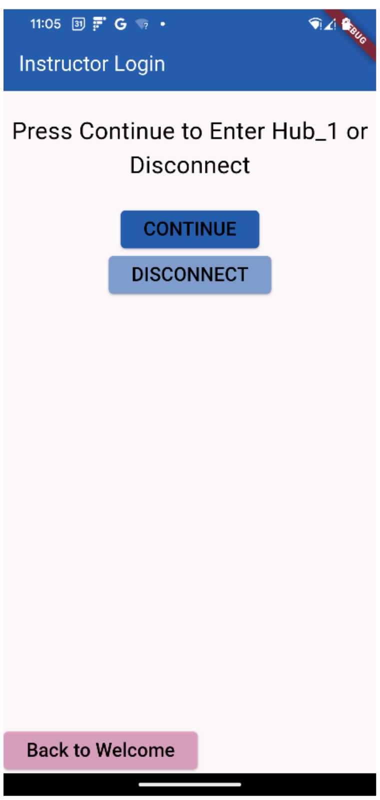


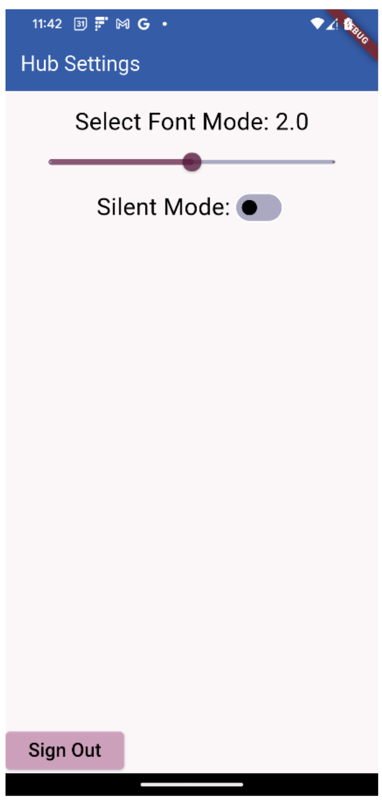
Instructor Path











Appendix E GPIO Pin Configurations

These figures are based on the ESP32-S3-WROOM-1 datasheet [1].

Strapping Pin	Default Configuration	Bit Value
GPIO0	Weak pull-up	1
GPIO3	Floating	–
GPIO45	Weak pull-down	0
GPIO46	Weak pull-down	0

Figure 20: Default configuration of strapping pins

Table 13: ESP32 pinout for control subsystem

Name	Pin Number	Function	Name	Pin Number	Function
GND	1	GND	IO14[1]	22	SPI; CS
3V3	2	Power supply	IO21	23	Unused
EN	3	Chip enable	IO47	24	Unused
IO4	4	Input; rotary encoder A	IO48	25	Unused
IO5	5	Input; rotary encoder B	IO45	26	NC; programming strapping pin
IO6	6	Input; clear button	IO0	27	Programming strapping pin
IO7	7	Output; vibration motor	IO35	28	Output; red LED set 5
IO15	8	Output; green LED set 1	IO36	29	Output; red LED set 4
IO16	9	Output; green LED set 2	IO37	30	Output; red LED set 3
IO17	10	Output; green LED set 3	I038	31	Output; red LED set 2
IO18	11	Output; green LED set 4	IO39[1]	32	Output; red LED set 1
IO8	12	Output; green LED set 5	IO40[1]	33	Output; yellow LED set 5
IO19	13	USB_D-	IO41[1]	34	Output; yellow LED set 4
IO20	14	USB_D+	IO42[1]	35	Output; yellow LED set 3
IO3	15	Output; blue indicator LED	RXD0	36	UART receiving pin
IO46	16	NC; programming strapping pin	TXD0	37	UART transmitting pin
IO9[1]	17	SPI; MISO	IO2	38	Output; yellow LED set 2
IO10[1]	18	SPI; SCK	IO1	39	Output; yellow LED set 1
IO11[1]	19	SPI; MOSI	GND	40	GND
IO12[1]	20	SPI; DC/RS	EPAD	41	GND
IO13[1]	21	SPI; RST			

[1]Differs from default function