Final Report
Angling Blinds

By Chaehee Lim, Kevin Choi, Kevin Yu

Team 20

TA: Dean Biskup

Dec 6, 2021

Abstract

This project aims to provide a user-friendly device that automates the turning of window blinds based on the brightness level of the interior. As the design focuses on matching the interior brightness level to the desired input of the user, the time that the room has the optimal brightness setting the user dictates is maximized. In this report, the project's design, requirements and verifications, and results are detailed. While there were shortcomings that affected the desired final product, we achieved a proof of concept design that allows for window blinds to rotate in quick intervals based on a photo sensor reading until the reading matches the desired brightness of a user.

Contents

1) Introduction

1.1) Problem

Due to the pandemic, there has been a large increase in people needing to work from home. However, it is a lot of work to get up and adjust your blinds throughout the day, and it is easy to neglect doing so while being preoccupied with work. Thus, many workers are not getting the proper natural sunlight they need, even though several studies have proven that natural sunlight provides many health benefits such as boosting Vitamin D, improving sleep, etc [1]. It is necessary for people to adjust their blinds to get the consistent level of sunlight they want but this is often forgotten.

1.2) Solution

To solve this issue, the project built blinds that adjust the angles to meet the desired level of brightness of the user, inputted through an app. A photo sensor detects how much sunlight is coming into the room and compares this value to the input of the user, adjusting the blinds through a stepper motor until the two values are close. There is also another photo sensor that detects how much sunlight is between the window and the blinds so that if there is not enough light outside to meet the expectations of the user, the app can notify the user. This way the user would be getting consistent sunlight throughout the day.

1.3) Functionality

For the project, there are four high level requirements to ensure a functioning product.

1. The blinds adjust to achieve the target brightness $\pm$ 200 lux.

   To make sure that at the end of the adjustment, the interior photo sensor reading and the desired input of the user need to be as close as possible since the entire point of the project is so that the inside brightness reaches the desired brightness. The 200 lux was a good minimum to ensure that the product worked. At the end of the project, however, the blinds adjusted to achieve the target brightness $\pm$ 100 lux.

2. The battery for the photo sensor devices must be able to last at least three months.

   The product needs to last a very long time so that it is worth using and since a barrel jack powered the motor driver PCB, to make sure that the photo sensor PCB lasts a long time, the requirement was set to last at least three months.

   Unfortunately, because the PCBs used Wi-Fi, compared to BLE according to the original design, the battery only lasted a couple of days and eventually ran out of battery soon after the demo day.

3. The blinds begin to adjust to the desired brightness within five seconds of the user inputting a value.

Ideally, we want the entire process of finding that desired brightness to be quick. After the user inputs the lux value they want in their room, we do not want them to wait long for the blinds to do its job. We felt that five seconds was a realistic goal to set but also not too slow. The finished product had the blinds beginning to adjust in less than five seconds of the user pressing the submit button.

4. The blinds finish adjusting within thirty seconds of starting.

Again, we do not want the user having to wait too long before their needs are met. And so, we wanted the blinds to stop adjusting after thirty seconds had passed since it started moving.

1.4) Subsystem Overview

1.4.1) Physical Design

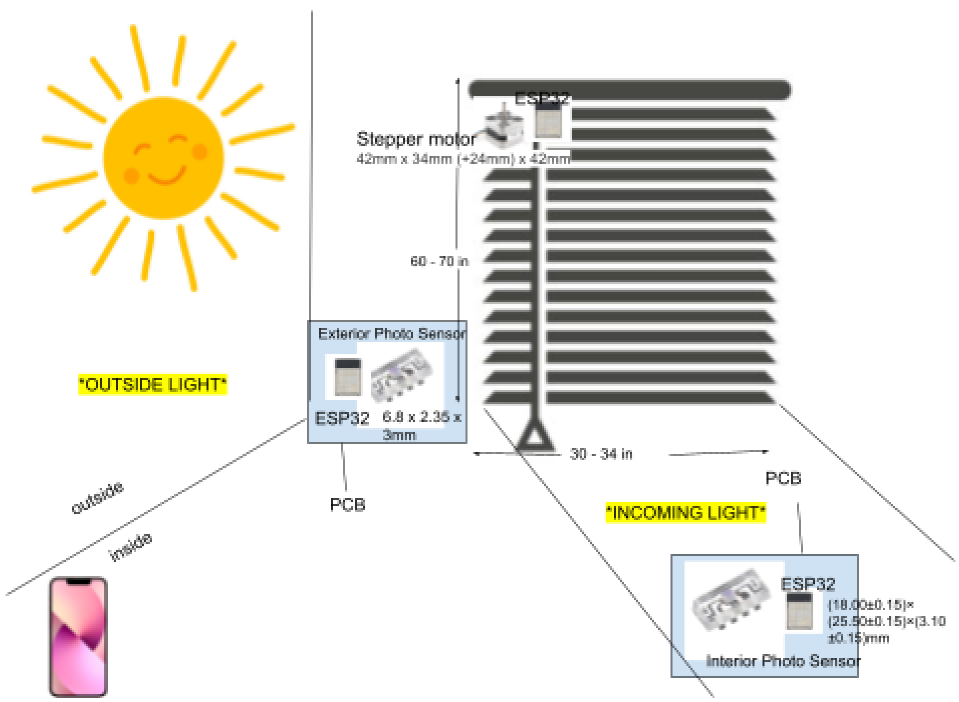

Figure 1: Physical Design

Figure 1 gives a visual representation of how we envisioned the product to work. There would be two photo sensors, one placed inside the room to read the incoming light and one placed between the blinds and the sun to read the amount of light outside. Then there would be a stepper motor attached to the blinds to rotate its angles with a microcontroller controlling the stepper motor. Lastly, the user would interact with all of these components through their phone.

1.4.2) Block Diagram

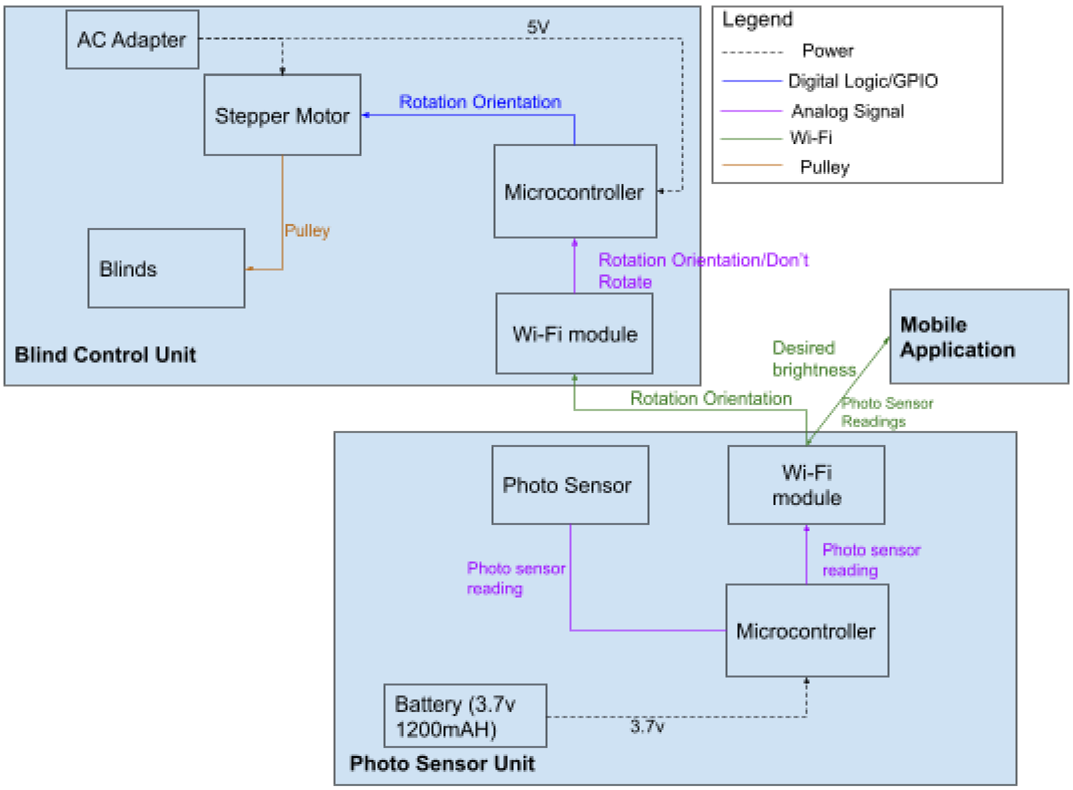

Figure 2: Block Diagram

The project has three subsystems connected through Wi-Fi: the Blind Control Unit, Photo Sensor Unit, and the Mobile Application, as shown in Figure 2. The Blind Control Unit uses a microcontroller to control the stepper that rotates the angle of the blinds and an AC adapter powers it. The Photo Sensor Unit also consists of a microcontroller but also has the photo sensor and a 3.7 V battery powers this unit. The access point is set to be the Interior Photo Sensor Unit, thus the Mobile Application communicates with the Photo Sensor Unit and the Photo Sensor Unit tells the Blind Control Unit the rotation orientation

Originally, the block diagram had the microcontrollers communicating through Bluetooth. Unfortunately, during implementation, we found that the microcontrollers had to be too close to each other for it to work. In Figure 1, the physical design had the two photo sensor PCBs somewhat far apart, where one was between the blinds and the window and the other was well within the room. Upon facing this problem, we switched to Wi-Fi.

Furthermore, the Blind Control Unit acted as the access point in the original design. But due to complications with getting the Motor Driver PCB to function, we had the interior photo sensor PCB act as the access point.

1.4.3) Logic

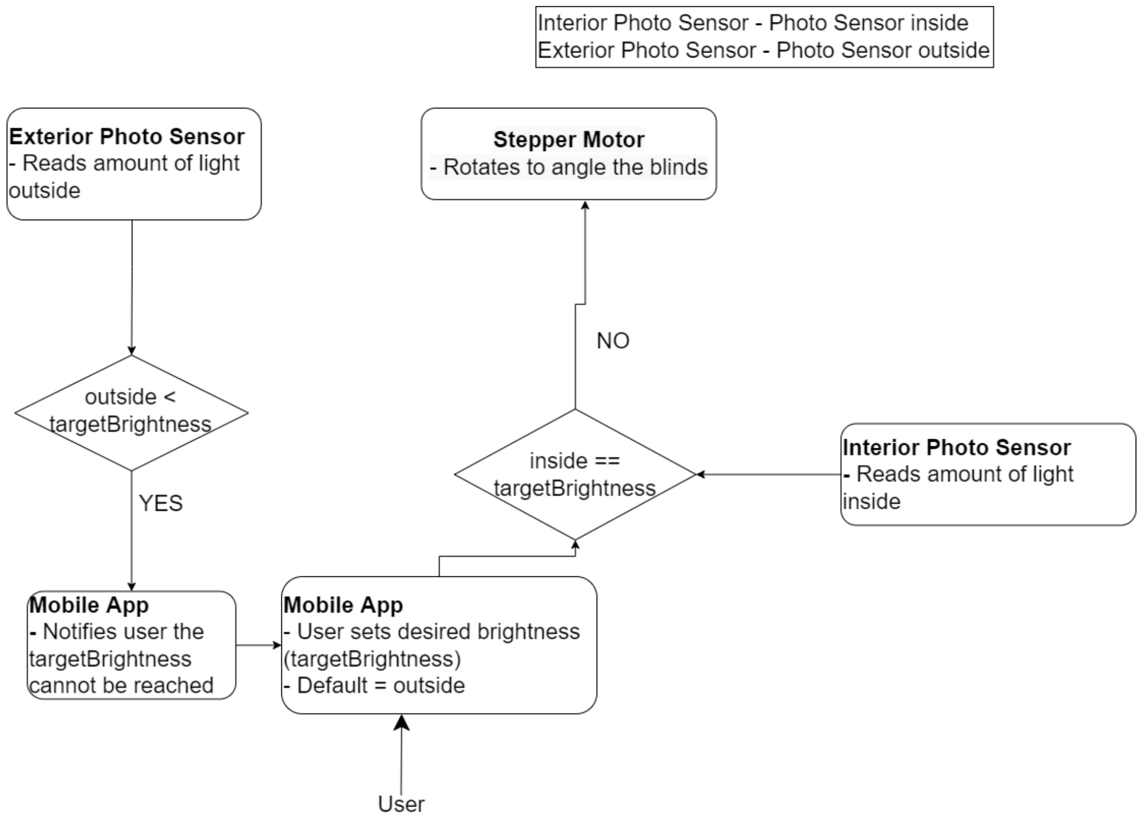

Figure 3: Flowchart

.

Following the flowchart in Figure 3, the user is to set their desired brightness in the mobile application. This value is then compared to the reading of the interior photo sensor. If the two values are not close, then the stepper motor rotates the angle of the blinds. If they are close, we do nothing and keep comparing the values until the two values are far from each other.

However, if there is not enough light outside, then the interior photo sensor reading will never be close to the requested brightness. Thus, there is an exterior photo sensor PCB reading the amount of light outside. If the amount of light outside is less than the user's desired brightness level, then the mobile notifies the user that there is not enough light outside.

For the final demo however, we had to remove the features of the exterior photo sensor and so, the left side of the flowchart was omitted.

2) Design

2.1) Blind Control Unit

2.1.1) Overview

The blind control unit controls the blinds, rotating the angles in the orientation the access point dictates it to. The subsystem includes a stepper motor, motor driver and an ESP32-WROOM-32D microcontroller. The stepper motor rotates the strings that control the blinds according to the instructions of the microcontroller. Figure 4 illustrates the schematic of the implemented PCB. This system is programmed through USB using a CP2102 USB to UART Bridge and powered by an AC adapter.

To know when and how to rotate, the Blind Control Unit periodically does an HTTP Get Request to get instructions from the access point the rotation orientation.

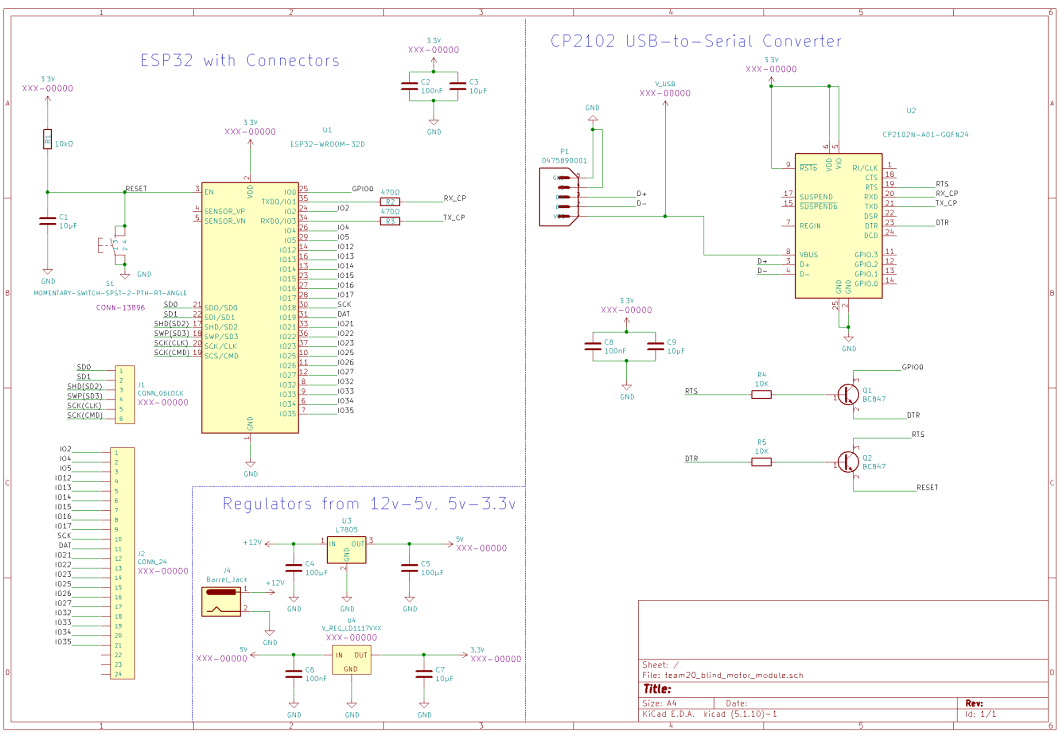

Figure 4: Motor unit schematic

2.1.2) Design Justification

We decided to use a stepper motor instead of a servo motor because this project required high position accuracy and reliability but did not necessarily need high top speed. We also chose to use the ESP32-WROOM-32D as the microcontroller because the original design required a

Bluetooth module and the microcontroller came with one built in. It also came with a Wi-Fi module built in, which proved to be useful when we found that to use BLE, the microcontrollers needed to be in close proximity [2].

2.1.2) Design Issues

After soldering the parts onto the PCB, we ran into an issue where the ESP32 microcontroller could enter the downloading mode, but was unresponsive when we attempted to upload the sketch. When trying to debug this issue, we found that some strapping pins had the wrong voltage in the boot mode [3]. And after some research, we concluded that this to be a hardware issue in which the chip was damaged when we desoldered it from the first-round PCB.

In an attempt to solve this issue, we cut off some wrong traces established from the footprint under the supervision of the TA and we manually set the voltage for the strapping pins voltage by connecting them directly to the power or ground. Moreover, we tried to adjust flash frequency and flash mode on Arduino IDE and compared booting status messages on the serial monitor. However, none of the attempts were able to identify the cause of the issue.

In the end, we could not resolve this issue and thus purchased an ESP32 Dev Module so that we can complete the project.

2.2) Photo Sensor Unit

2.2.1) Design Overview

The photo sensor unit consists of a photo sensor, a microcontroller, and a battery. There are two of these units spread out across the room. The microcontroller in this subsystem is also programmed in a CP2102 USB to UART Bridge. The interior photo sensor pcb connects to the blind control unit via Wi-Fi where the blind control unit periodically reads the HTTP Get request containing the rotation orientation [2].

The first photo sensor is to be well within the room to abstain from the general brightness of the room itself to make sure that the desired amount of sunlight is entering the room. The second photo sensor reads the output from the outside window to record the maximum brightness so that the user knows the maximum sunlight that can be brought inside the room. Both of these readings would be sent via Wi-FI through the microcontroller to the mobile application.

These units are crucial in monitoring the room and the brightness outside so that the blinds can adjust when necessary. The interior photo sensor acts as the access point for the Wi-Fi network while the exterior photo sensor does an HTTP Post request to send its readings.

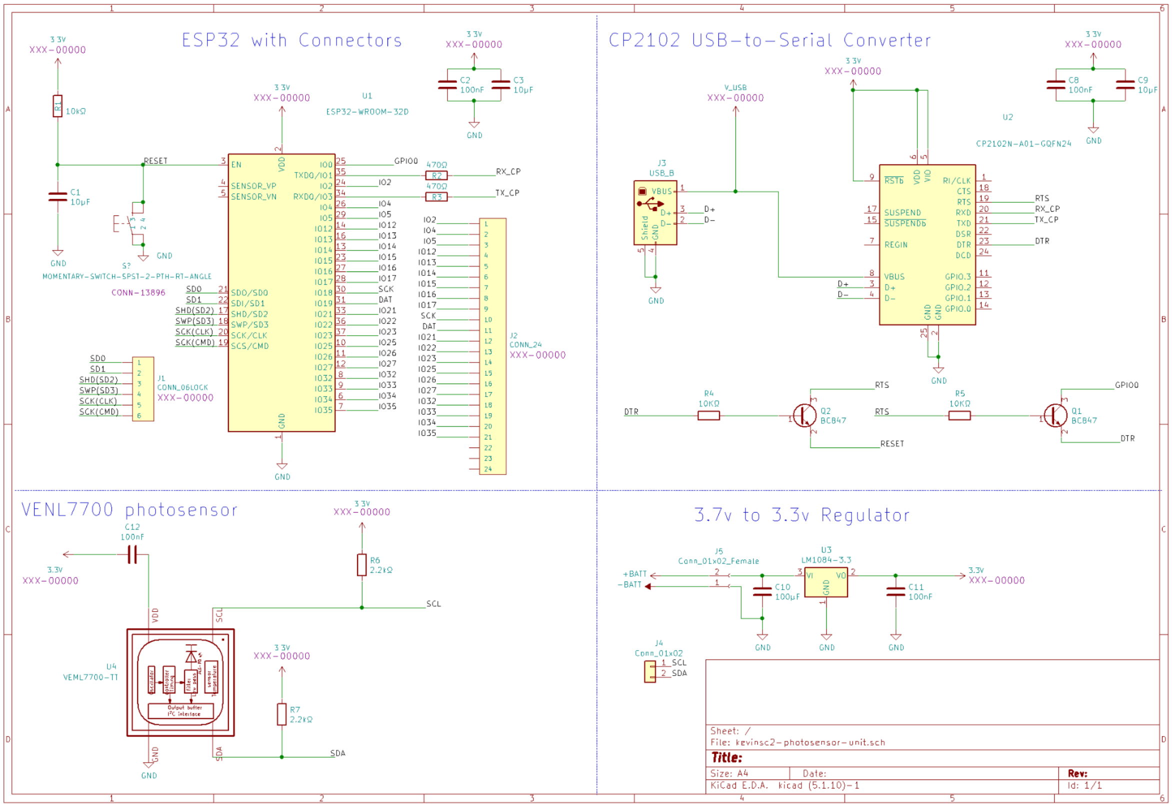

Figure 5: Photo sensor unit schematic

2.2.2) Design Justification

We decided to again use the ESP32-WROOM-32D because of its built-in Bluetooth/Wi-Fi module. To ensure accurate brightness readings, the VEML7700 ambient light sensor was chosen due to its high ambient light range from 0 to 120,000 lux [4]. This unit is powered by a 3.3 V voltage source stepped down using a linear regulator from a 3.7 V 1200mAH battery [5].

2.2.3) Design Issues

Initially, we had issues with the LD1117 regulators failing to correctly provide a consistent output 3.3 V from the 3.7 V battery voltage. And after some research, and learning that other people experienced the same issue, we decided to switch to TC1262 which is more stable and generally more reliable with the ESP32 series [6].

Two days before the demo, the interior photo sensor stopped working. It faced a similar issue that the motor driver PCB faced. From doing some research, we deduced that the issue came from a brownout due to the switch from BLE to Wi-Fi since Wi-Fi consumes a lot more power than BLE. Thus, we had to work around this issue and decided to remove the exterior photo sensor features since the interior photo sensor was a more core part of the project.

And since WI-Fi consumes a lot more power than BLE, the battery lasted a very short time and the battery drained for the interior photo sensor very quickly most likely due to it being the access point.

2.3) Mobile Application Unit

2.3.1) Design Overview

The application displays two photo sensor datas and has an input for the user to enter their desired brightness. If the exterior photo sensor data is less than what the user has inputted, then the app notifies the user. To use the application, the user must connect their device to the wireless network of the access point. From there, the application receives the two datas using an HTTP Get request and sends the input as an HTTP Post request. This is where the user mainly interacts.

Figure 6: Mobile App when user puts in value greater than exterior data

2.3.2) Design Justification

For the app, we used Ionic Platform, mainly because we were all familiar with React and it supported exporting to both Android and IOS. It proved to be a good platform as we could easily code the app to the way we felt best showcased the project [7].

We wanted to keep the app simple and just have the minimum features to satisfy the functions.

2.3.3) Design Issues

After we decided to get rid of the exterior photo sensor PCB, the features that came with the exterior photo sensor on the app also had to be removed. Nonetheless, the main function of moving the blinds using the input of the user did not change.

While initially we wanted the app to be a mobile app, because we ran out of time, during the demo, we had to use the app as a web app rather than exporting it to be a mobile app on the phones.

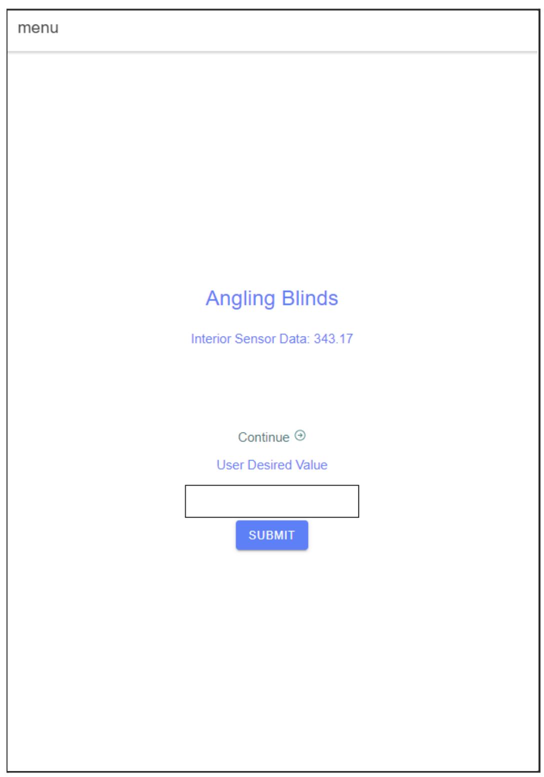

Figure 7: Mobile App for the demo

3) Cost and Schedule

3.1) Cost

3.1.1) Labor Cost

(50/hour) x 2.5 x (60 hours to complete) = $7500/member
$7500/member x 3 members
= $22500 for the entire project for team members
(50/hour) x (40 hours to complete) = $2000
$2000 for machine shop labor

3.1.2) Part Cost

Item	**Quantity**	**Cost**
Blinds	1	17.59
esp32-wroom-32	3	$13.50
Big Easy Driver	1	$19.95
Tactile switches	6	$5.70
3.7 V batteries	2	$19.90
TC1262_33	3	$1.89
LM7805	1	$3.05
Barrel Jack Connector	1	$1.50
10k resistors	3	$3.33
2.2k resistors	4	$1.48
100nF capacitors	3	$3.30
1uF capacitors	5	$1.50
0.33uF capacitors	1	$0.30
10 pin connectors	1	$0.80
6 pin connectors	2	$0.30

JST connector	2	$1.50
VEML 7700 TT	2	$4.26
12V power Supply	ordered	$9.00
Total		$108.85

3.1.3) Total Cost:

$22500 + $2000+ $108.85= $24608.85

The total cost for this product would be $24,608.85

3.2) Schedule

Week 9/27 -

(All) - Prepared for the Design Review, finishing up the design document and asking our TA to look over it.

(Kevin and Kevin) - Began designing the PCB schematics for the first round of PCBs.

Week 10/4 -

(All) - Finish PCB designs to be sent

(All) - Order the parts

Week 10/11 -

(All) - Figure out the what platform to use for the app

(All) - Created general design for the app

(Kevin and Kevin) - Start working on general UI

Week 10/18 -

(All) - Continue working on the app

(All) - pick up the parts that is coming in

Week 10/25 -

(All) - Solder the parts

Week 11/1 -

(Kevin and Kevin) - Redesign PCBs so that the sizes match so that we can actually use and test the PCBs

Week 11/8 -

(All) - Solder the parts to new PCBs

(Kevin and Kevin) - de-solder the parts of the old PCBs

(Kevin Yu) - programming microcontroller for photo sensor

(Chaehee) - implement BLE

(Kevin Yu) - debug Motor Driver PCB

(Chaehee and Kevin Choi) - debug BLE communication

Week 11/15

(Chaehee and Kevin Choi) - implement Wi-Fi, decide the data flow with the motor driver currently not working

(Kevin Yu) - connect Dev Module to motor

Week 11/22 -

FALL BREAK

(Kevin Choi and Chaehee) - connect Motor Driver PCB with Access Point (Interior Photo Sensor)

(Kevin Choi and Chaehee) - implement in code so the motor stops at max and min steps to not break blinds

Week 11/29 -

(All) - Figure out what to do about one of the photo sensor breaking

(Chaehee) - Fix app according to adjustment of no exterior photo sensor

(All) - Final adjustments and tests

(All) - Figure out demo, have demo

(All) - Start Final Report

(All) - Work on Presentation, Have mock Presentation

Week 12/6 -

(All) - Finish presentation, rehearse presentation, have final presentation

(All) - Work on Final Report

4) Requirements & Verification

4.1) Blind Control Unit

Requirements	Verification
1. The stepper motor rotating adjusts the angles of the blinds in each orientation. 2. The motor stops before rotating beyond the set maximum total number of steps and under the set minimum total number of total steps to prevent damage to the blinds. 3. The motor starts moving within 3 seconds of the ESP32 notifying it to.	1. Verification Process for Item 1: a. Have the stepper motor rotate to the right and this adjusts the angles towards closing one way. b. Have the stepper motor rotate to the left and this adjusts the angles towards closing the other way. 2. Verification Process for Item 2: a. Have the blinds in a fully open orientation b. The microcontroller tells the motor to rotate to the left continuously. c. The motor should stop at the set maximum total number of steps. d. The microcontroller tells the motor to rotate to the right continuously e. The motor should stop at the set minimum total number of steps. 3. Verification Process for Item 3: a. Program the ESP32 to tell the motor to start rotating in one direction X number of steps. b. Start a timer. c. Stop the timer when the motor starts moving. d. Program the ESP32 to tell the motor to start rotating in the other direction X number of steps. e. Start a timer. f. Stop the timer when the motor starts moving. g. Ensure both times, it took less than 3 seconds. .

	Rotates? (Trial 1)	**Rotates? (Trial 2)**
Rotate left	Yes	Yes
Rotate right	Yes	Yes

Table 1: Results of verification of requirement 1

	Stops? (Trial 1)	**Stops? (Trial 2)**
Rotate left	Yes	Yes
Rotate right	Yes	Yes

Table 2: Result of verification of requirement 2

	Time (Trial 1)	Time (Trial 2)
Rotate right	0.33s	0.31s
Rotate left	0.31s	0.34s

Table 3: Result of verification of requirement 3

4.2) Photo Sensor Unit

Requirements	**Verification**
1. The microcontroller can report the photo sensor output. 2. The photo sensor must be able to accurately read the brightness of the area with a bound of ±200 lux 3. This unit must communicate to the blind control unit using a wireless network in at least 0.5 second. .	1. Verification Process of Item 1: a. Cover the photo sensor. b. The Microcontroller should report 0 lux ±200 lux. c. Uncover the photo sensor. d. The Microcontroller should report not 0 assuming it is not completely dark in the room. 2. Verification Process of Item 2: a. Have both photo sensor units right next to each other b. Obtain the lux reading of both photo sensor units c. Check if they are within 200 lux of each other 3. Verification Process of Item 3: a. Have the photo sensor unit send a print message to the blind control unit b. Look at the time duration between each print message sent to the blind control unit c. Verify if the duration is less or equal to 0.5 seconds

	Serial Output (Trial 1)	Serial Output (Trial 2)
Covered	10.71	8.25
Uncovered	465.2	482.2

Table 1: Result of verification of requirement 1

	Reading (Trial 1)	Reading (Trial 2 - covered)
Photo sensor 1	486.3	20.5
Photo sensor 2	471.2	14.51

Table 2: Result of verification of requirement 2

	Time
Trial 1	0.32s

Trial 2	0.28s

Table 3: Result of verification of requirement 3

4.3) Mobile App Unit

Requirements	**Verification**
1. The desired brightness set by the user is communicated properly to the Photo Sensor Unit's microcontroller. 2. The user sees the maximum brightness outside as read by the photo sensor outside through the application. 3. If there is not enough light outside to match the desired brightness level, the application lets the user know through a text (ex. "There is not enough light outside).	1. Verification Process for Item 1: a. Set the brightness to no light in the app. b. The microcontroller prints the value coming from the application as 0 lux c. Set the brightness to 10,000 lux d. The microcontroller prints 10,000 lux 2. Verification Process for Item 2: a. Manually send the 3 different brightness measurements at different times in the day (brightest, middle, darkest) from the microcontroller of the photo sensor unit. b. The app should return the same value as the maximum brightness outside. 3. Verification Process for Item 3: a. Set the desired brightness at the darkest setting at the peak of the day (12 pm - 3 pm) b. The text should not show up. c. Set the desired brightness as a higher lux value than the maximum brightness around 4 pm and around evening (past 9 pm) the application should have the text "There is not enough light outside".

Input	**Serial Output (Trial 1)**	**Serial Output (Trial 2)**
0	0	0
10,000	10,000	10,000

Table 1: Result of verification of requirement 1

Requirements 2 and 3 could not be met at the demo because one of the photosensor PCBs stopped functioning. Thus, we could only show one of the photo sensor readings and we chose it to be the interior. And since we did not know what the outside brightness is, we could not say whether there was or was not enough light outside.

5) Conclusion

5.1) Accomplishments

We were able to successfully create a project that automates the rotation of window blinds based on the interior brightness of the room. The window blinds were able to correctly turn in both orientations from input from the motor driver that was influenced by the ambient light sensor on the photo sensor PCB. Communication between all of the components was possible with Wi-Fi, having the photo sensor PCB act as a client for the ESP32 dev module and mobile application to connect to. The mobile application displays the current brightness level of the room in real-time as well.

Also, the stepper motor correctly starts rotating when detecting that a desired brightness of the user is outside ±100 lux of the actual brightness of the room, constantly adjusting until either the brightness of the room is correctly adjusted or 30 seconds has elapsed since the original command in the case that the desired room brightness is not feasible with blind rotation.

5.2) Uncertainties

There are uncertainties with the current iteration of the project that need to be addressed for future iterations. One uncertainty is one of the photo sensor PCBs that stopped working due to a hardware issue. As we cannot measure the brightness of the exterior photo sensor as a result, we do not have functionality for notifying the user if the desired input is not feasible.

Another uncertainty is the brownout issue with the motor PCB that caused it to be not functional and therefore needed us to use an external ESP32 dev module. Since the motor PCB was supposed to power both the motor driver and the microcontroller through a 12 V barrel jack and voltage regulators, the motor PCB malfunctioning caused issues with powering these components. We needed to power the dev module using a USB port to a computer and the motor driver needed to be powered using a power supply attached to a breadboard providing 12 V. Addressing the current issues with the motor PCB will allow for a much simpler setup where external components do not need to be used to supplement the functionality of the stepper motor.

5.3) Future Work / Alternatives

There are some adjustments that the project needs to make before being considered to be fully functional to the requirements and verifications. The current state of the project only has one working photosensor unit which is used to measure the brightness of the interior. For the future, we will ensure that both photo sensor units will be functional to include the currently omitted features of the exterior photo sensor. Also, there are other issues that are present in the current iteration of the project, namely wiring and positional errors for the PCB. We had errors in the KiCAD schematics that forced us to use an external wire to connect two components originally intended to be connected through the PCB wire tracing. Also, the antennas for the microcontrollers should have been positioned outside the physical PCB due to potential grounding issues that may have contributed to the brownout error.

For future design considerations, we would also add a position sensor to track the current angle of the blinds. The stepper motor occasionally slips and as a result does not consistently rotate the blinds the right amount of steps. Adding a position sensor would allow for consistent rotation of the window blinds as intended by the stepper motor without risk of slipping. Also, we would like to implement a chassis for the PCBs, as currently the PCBs are completely exposed visually during actual use and we would like to change this for the finalized product.

5.4) Ethical Considerations

This project has potential ethics and safety issues that may conflict with the IEEE Code of Ethics, namely the safety and privacy of the user [8]. For instance, the user could get their finger, hair, etc. stuck in the motor area. As a preventative measure, the implementation of the stepper motor is designed to be turning at a low speed to ensure user safety in a potential case of physically interacting with the motor. We would also install the motor so that it is out of reach for small children.

The user must be okay with an automated system opening and closing their blinds as this can be a privacy issue. As the designers for this project, it is important to be transparent about the potential privacy issues of the project and to address ways to ensure the privacy of a sure. As the user will be controlling the behavior of the motor as it controls the blinds through the application, the user must understand the possibility of their space being seen as a result of the system adjusting the brightness level of their space. The mobile application will also have a manual setting where the user will be able to fully control the orientation of the blinds to respect the privacy of the user.

We will accept and apply honest criticisms of the product we produce as well as credit the proper contributions of others. As stated by the IEEE Code of Ethics, we will respect every person and not discriminate regardless of their race, religion, gender, disability, age, national origin, sexual orientation, gender identity, or gender expression [8]. We will also not engage in harassment and avoid causing injuries to others, property of others, reputation, or employment through any form of abuse or malicious actions [8].

References

[1] Garone, Sarah. “11 Things to Know about Natural Light and Your Health.” Healthline, Healthline Media, 11 Aug. 2020, www.healthline.com/health/natural-light-benefits.

[2] ExpressIf Systems. (n.d.). ESP32-WROOM -32D & ESP32-WROOM-32U Datasheet. ExpressIf. Retrieved September 29, 2021, from https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32d_esp32-wroom-32u_datasheet_en.pdf.

[3] Santos, S., 2021. ESP32 Pinout Reference: Which GPIO pins should you use? | Random Nerd Tutorials. [online] Random Nerd Tutorials. Available at: <https://randomnerdtutorials.com/esp32-pinout-reference-gpios/> [Accessed 9 December 2021].

[4] High Accuracy Ambient Light Sensor With I2C Interface. Vishay Semiconductors. (n.d.). Retrieved September 30, 2021, from https://www.vishay.com/docs/84286/veml7700.pdf.

[5] Random Nerd Tutorials. 2021. *ESP32 HTTP GET and HTTP POST with Arduino IDE | Random Nerd Tutorials*. [online] Available at: <https://randomnerdtutorials.com/esp32-http-get-post-arduino/> [Accessed 9 December 2021].

[6] 500mA Fixed Output CMOS LDO. Retrieved December 9, 2021,from https://ww1.microchip.com/downloads/en/DeviceDoc/21373C.pdf.

[7] “Open-source UI toolkit to create your own mobile or desktop apps: Ionic Documentation,” Open-Source UI Toolkit to Create Your Own Mobile or Desktop Apps | Ionic Documentation. [Online]. Available: https://ionicframework.com/docs. [Accessed: 09-Dec-2021].

[8] “IEEE Code of Ethics.” IEEE, www.ieee.org/about/corporate/governance/p7-8.html.