Automatic Light Switch

By

Sangsun Lee

Ruize Sun

Andrew Kim

Final Report for ECE 445, Senior Design, Spring 2025

TA: Eric Tang

May 6th, 2025

Project No. 71

Abstract

Our project is an Automatic Light Switch system designed to be mounted over existing dimmable manual light switches, enabling users to remotely control their lighting with ease and flexibility. The device connects to both a mobile app and a web-based interface, allowing users to adjust light levels in real time from a distance, thanks to reliable Wi-Fi connectivity. One of the key strengths of our design is that it preserves the original manual switch functionality while introducing seamless remote control. Throughout testing, the system consistently demonstrated accurate dimming control, quick responsiveness through both platforms, and stable wireless performance. The lead-screw mechanism effectively adjusts the physical dimmer without altering or damaging the existing switch, making the device non-invasive and ideal for renters or shared living environments. Overall, this solution offers a practical, and user-friendly approach to upgrading traditional lighting systems into smart, remotely controllable light switch.

Contents

1. Introduction

Many buildings and rooms still rely on traditional light dimmer switches that require users to manually slide a knob up or down to adjust the brightness of a light bulb. While smart light switches offer improved convenience, and automation, upgrading to these systems usually involves removing the original switch and installing a smart replacement. This process often requires rewiring and electrical modifications, which can be costly, complex, and time-consuming. For renters or individuals living in temporary accommodations, these upgrades are usually not feasible due to lease agreements or restrictions that prohibit making permanent changes to the property's electrical system.

This limitation creates a significant barrier for a large segment of the population who would otherwise benefit from smart lighting solutions. Although some retrofit products exist to add smart functionality, many are either expensive, difficult to install, or limited in features, making them inaccessible or unattractive to the average user. These challenges highlight a growing need for innovative solutions that can provide smart lighting control without requiring structural modifications or specialized technical skills. A device that can be easily mounted over an existing dimmer switch—without any wiring changes—could offer a simple, affordable, and renter-friendly alternative to traditional smart lighting systems.

1.1 Solution

Many renters and homeowners are interested in smart lighting solutions, but they often encounter challenges such as complex wiring requirements, the need for permanent modifications, or restrictions imposed by landlords. Most traditional smart switches require rewiring of the existing electrical system, which can be both technically demanding and impractical for those without electrical experience.

This automatic light switch is designed to mount over an existing dimmable light switch without any wiring modifications. It allows users to:

1. Manually control their lights just like a traditional switch, with the added ability to adjust light levels through an app, web interface, or voice commands for greater convenience and customization.
2. Seamlessly integrate with Wi-Fi, allowing wireless operation over long distances without the need for additional hubs or complex setup.

This automatic light switch is unique and beneficial to the consumer because

1. Easy Installation: No tools or rewiring required, making it renter-friendly.
2. Convenience – Adjust lights with voice commands or through the app from anywhere.
3. Customization – Precise brightness control for different moods and activities.
4. Accessibility – Ideal for users with mobility limitations, as lights can be controlled remotely.

The key features of this automatic light switch is that

1. Voice assistant compatibility for hands-free operation.
2. Wi-Fi enabled mobile app and web-interface with a smooth brightness button.

3. No-wiring, no-drill needed, just mount it over any dimmable light switch.

This automatic light switch offers consumers a hassle-free way to modernize light control, making smart home upgrades accessible to everyone.

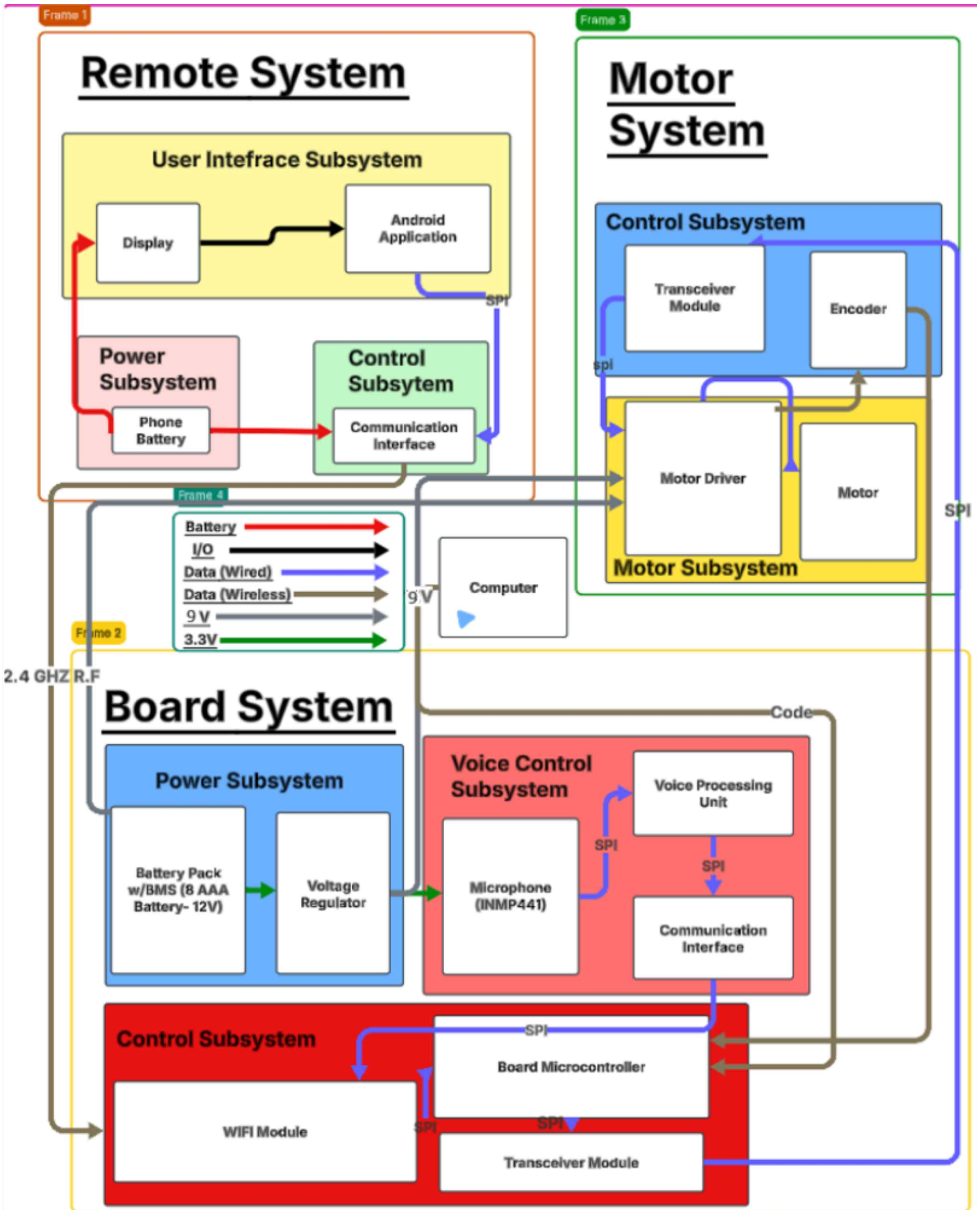

Figure High-Level Diagram of the Automatic Light Switch

In our automatic Light Switch, we have a total of 3 System, Remote system, Board System, Motor System.

Remote system of the automatic light switch involves creating a smart lighting control system using an Android application and a web interface built with Android Studio and WordPress, both of which communicate with an ESP32-S3 microcontroller via Wi-Fi. The system allows users to select between five light levels: ON, OFF, Night mode, Reading mode, Bright mode. The Android app and the web interface form the User Interface Subsystem of the remote system. When the user selects a light level, the Android application sends control signals through SPI to the Transceiver module within the Control Subsystem in the Remote system. The Communication Interface then relays these commands to the Wi-Fi module of the ESP32-S3 using UART. The ESP32-S3 processes the command and send it to the Motor system which actuates the light switch accordingly.

Board System acts as the central hub for processing and power distribution. It receives voice commands through the INMP441 microphone and processes them using digital signal processing before passing them to the ESP32-S3 microcontroller. The board is powered by a 9V battery, which supplies power to both the ESP32-S3 and the motor driver module using a voltage regulator. The ESP32-S3's onboard Wi-Fi module enables long-distance communication between the board system and the remote user-interface subsystem, which includes voice control, mobile app, and web interface options. When the user sends a command through any of these interfaces, the input is transmitted via Wi-Fi to the board system. The ESP32-S3 interprets the input and sends control signals to the motor system. This system then actuates the physical light switch accordingly.

Motor System receives power from the 9-volt battery housed in the Board System, which is regulated using voltage regulators to ensure proper operating conditions for the stepper motor. This setup allows the motor to be safely and reliably powered without overloading the system. The ESP32-S3 microcontroller in the Board System receives the signal from the user-interface subsystem in the Remote system and sends control signals to the Transceiver Module located within the Motor System. These signals determine the direction and extent of the motor's rotation. Based on the received instructions, the motor performs precise movements that physically adjust the attached light switch to match the user's input. Whether turning the switch on, off, or adjusting to a specific light level, the motor acts in direct response to the processed signal.

2 Design

This project presents a complete intelligent switching system that integrates software, hardware and mechanical components. The system is capable of wirelessly controlling a 5 Level mechanical sliding switch using both a mobile application and a web interface. The design involves 3 system: remote system, board system and control system.

In the remote system, one of the subsystems was an application design. An Android based mobile APP was developed using Android studio, XML and Java. The app presents a simple user interface with button labeled “ON”, “OFF”, “Night mode”, “reading mode”, and “bright mode”, each representing a switch level. When the level is selected, the app sends an HTTP Post request to the ESP32’s IP address over a local Wi-Fi network. The App allows real time feedback for successful switching, and error message will be displayed for failed switching. Another subsystem was a web-interface design. A webpage was written in HTML, CSS and JavaScript, and include buttons for each of the 5 levels. The operating Mechanism of the webpage is like app. When a button is clicked, a JavaScript functions send an HTTP request using fetch to the ESP32 backend. User can access the webpage through any device connected to the same network such as Laptop. Therefore, the dual-interface design ensures continuous operation even if one channel like app fails due to the connectivity or device issues.

In the board system, one of the subsystems is embedded hardware design. The ESP32 serves as the core controller of the overall system. It connects to Wi-Fi networks and runs a web server that listens for control commands sent from either mobile app or webpage. Upon receiving a level command, the algorithm inside ESP32 programming will calculate the required number of motor steps and directions, driving the stepper motor through A4988 (Stepper motor driver). All logic includes web server, motor control algorithm, Wi-Fi is implemented using the Arduino framework in C++. The ESP32 was powered using battery, with a 5V voltage regulator. Another subsystem is PCB design. A custom PCB was designed to integrate all hardware components, including the ESP32, A4988 (Stepper motor driver), voltage regulator and necessary electronic components. In the PCB, the LDO regulator and decoupling capacitors provide stable and enough voltage power supply for ESP32. The GPIO pins from the ESP32 are routed to control MS1, MS2, MS3, STEP, DIR of A4988 stepper motor driver. A 9V battery is connected to VMOT which is the used to provide the power supply for the stepper motor. The VDD in stepper motor driver is logic power supply. We also need to adjust the current limit potentiometer on the stepper motor driver to protect our motor as the Max current limit is 800mA. Furthermore, additional headers are provided for battery input, motor connection and USB flashing access.

In the control system, the core of the design is mechanical arm that clamps the sliding switch and push it vertically. The switch has 5 discrete level, spaced evenly, the mechanical arm ensures the switch move to the fixed position forward and backward. The ESP32 will execute the command from the Webpage and app and send the signal to motor, ensuring the light switch can be pushed to the expected position. To control the motor, an A4988 motor driver is used. The ESP32's GPIO pins are connected to the STEP and DIR pins of the A4988, allowing for precise step and direction control. Appropriate capacitors are added across VMOT and GND to stabilize power. Micro stepping configuration is set by connecting MS1–MS3 pins as needed.

After the design review, we have made significant modifications to the drive mechanism, replacing the DC motor with a stepper motor. Initially, the system used a DC motor due to its simplicity and lower cost. However, during early testing, it became apparent that the DC motor lacked the necessary precision required to accurately stop at the five discrete switch positions. To address this issue, we switched to a stepper motor, which provides precise angle control and does not require feedback for accurate position. This modification improved system reliability and accuracy and reduced the complexity of the logic. As a result, the overall performance and stability of the system was greatly enhanced.

During the hardware design phase, we encountered several key issues related to motor driver protection and power regulation. First, the VMOT pin of the motor driver requires a large decoupling capacitor to absorb voltage spikes and prevent driver damage. We initially selected a single large capacitor but found that its voltage rating was insufficient for our 9 V system. To address this, we used multiple smaller electrolytic capacitors with higher voltage ratings in parallel. This not only solved the voltage rating issue but also improved transient response, resulting in better power stability. Second, we found that connecting or disconnecting the stepper motor while power was applied led to system instability. This oversight resulted in the failure of several motor driver modules during early testing, requiring multiple replacements. To prevent further damage, we revised our procedure to ensure that motor connections are only made when the system is powered off. Third, we initially neglected to adjust the current limit via the VREF pin, which caused the stepper motor to draw more than the recommended 800 mA. This excessive current likely contributed to overheating and further damage to the stepper motor. After identifying this oversight, we calculated the appropriate VREF value based on the motor's current requirements and adjusted the current limit potentiometer to ensure proper current limiting. Finally, when using a 9 V battery with an L7805 regulator, we found that the output dropped to around 4.6 V as the battery voltage fell to 8.2 V under load. This was below the MCU's required input and caused startup failures. As a temporary solution, we purchased additional batteries to ensure adequate voltage. For future versions, we plan to switch to a buck converter for better performance with battery power.

2.1 Design Analysis

We have predicted the maximum current that can flow through the PCB to prevent any power wastage or shorting issues. If the current is too strong, there can possibly be overheating issues and the soldering of the PCB melting down due to high heat. The melting down of soldering can lead to a chain of issues such as disconnection of circuitry or short circuiting.

The maximum current flowing through the PCB can be calculated using the following equation:

$$A = T \times W \times 1.378 \, [\text{mils} / (\text{oz} / \text{ft}^2)] \tag{2.11}$$

Where

A - Cross-section area

T - Trace thickness [mils2]

W - Trace width [oz/ft2]

To prevent any high voltage issues, we added capacitors to adjust the voltage in certain parts of our circuit. This was done by measuring parts of our circuit and testing the resulting voltage after placing capacitors.

The number of steps required to achieve the desired rotational angle is given as follows:

$$N = (\theta / 360) \times S \times M \tag{2.12}$$

Where

N - number of steps to move

S - number of full steps per revolution

θ - the desired rotation angle (in degrees)

M - the micro stepping factor, determined by MS1-3 setting (see Table 1)

Using this formula, we can adjust the Arduino code to change the number of micro steps the motor will take. This allowed us to achieve the wanted number of rotations for the motor and control the distance the switch moves accordingly.

To achieve precise positioning of the motor, the stepper motor needs to complete S full steps (200 steps in this project) for each revolution. By setting the micro step control pins MS1, MS2, and MS3, each full step can be subdivided into multiple smaller micro steps. The corresponding relationship is shown in the following table:

MS1	MS2	MS3	Microstep Resolution	Excitation Mode
Low	Low	Low	Full step	2 Phase
High	Low	Low	Half step	1-2 Phase
Low	High	Low	Quarter step	W1-2 Phase
High	High	Low	Eigth step	2W1-2 Phase
High	High	High	Sixteenth step	4W1-2 Phase

Table 1. Micro stepping resolution truth table

The maximum current flowing through the stepper motor is 800 mA. To protect our stepper motor from overcurrent damage, we must configure the current limit accordingly. This is achieved by adjusting the reference voltage (VREF) on the A4988 stepper driver. The relationship between the current limit and the reference voltage is given by:

$$I_{MAX}= \frac{V_{REF}}{8 \cdot R_{CS}}$$

where R_{CS} is the current sense resistor on the driver board. For A4988 modules manufactured, $R_{CS}= 0.068\,\Omega$. Substituting $I_{MAX}= 0.8$ A, we get:

$$V_{REF}= 8 \cdot I_{MAX} \cdot R_{CS}= 8 \cdot 0.8\text{ A} \cdot 0.068\,\Omega = 0.4352\text{ V}$$

Therefore, to limit the current to 800 mA, the voltage on the VREF pin should be adjusted to approximately 0.435 V.

3. Design Verification

All high-level and subcomponent requirements are shown in Appendix A. The verification processes mainly consisted of testing the final product a number of times and recording the resulting data.

Additionally, we have measured the actual angle compared to the predicted angle and made sure that the angle was within the error range. The angles we have measured are given below:

Direction	MS1	MS2	MS3	Expected Angle (°)	Measured Angle (°)
1 -> 3	High	High	Low	1080	1069
3 -> 2	High	High	Low	540	548
2 -> 5	Low	Low	High	1620	1635

Table 2. Experimental results of step motor control

In addition to verifying whether we can control the light switch at a distance of 30 meters as the high-level requirement, we have also measured the maximum distance we can send signals to control the switch. We have conducted a total of 5 trials to measure the average maximum distance it can send signals through.

Trial	Distance Measured (m)
1	90
2	95
3	85
4	100
5	95

Table 3. Experimental results of maximum distance

Every trial, we repeatedly sent signals to see the maximum distance we can send signals. Upon failure, we recorded the distance where the signal transfer was previously successful. After the 10 trials we have taken, the average was calculated to show the reliable distance a user can use the light switch with a low chance of failure.

We have also sent signals through obstacles to verify the reliability of data transfer in special cases. The obstacles we have recorded that we can send signals through are given as follows:

Obstacles	Approximate Distance	Verification Status (Y/N)
No obstacles (outside)	100	Y
Single door	40	Y
Wall	30	Y
Double door	30	N
Different floor (1 floor difference)	30	Y

Table 4. Experimental results of obstacles measured

The signal transfer through obstacles was also measured through a similar process, but instead of measuring distance, we recorded the obstacles we have attempted to send signals through. Obviously, locations with no obstacles worked with no issues. Single door and walls in between sending signals also worked with great precision, as it worked all of the times. Locations where double doors were installed did not allow stable connection between the ESP32 WiFi module and user interface, which is why we concluded it failed to pass the verification. Finally, testing the light switch at a different floor worked most of the times, with one trial failing to pass, but working for the majority (4 out of 5) of the trials. This data allows the user to understand the specific places and situations where the light switch might fail.

The time taken from the user to send signals until the MCU receives and controls the motor was measured through a stopwatch. The stopwatch was initiated when our groupmate sent a signal through both the application and website, and stopped when the motor started to move the light switch. 5 trials were attempted for each the application and website to measure the average time taken to transfer signals.

Trial	**Time Taken (s)**
1	0.56
2	0.44
3	0.51
4	0.50
5	0.48
Average	0.50

Table 5. Experimental results of time taken to transfer signals

From this data, we can conclude that the data transfer is very reliable. The distance we can send signals is especially high since our product uses the WiFi module, so the reliability of sending signals is dependent on how reliable the WiFi connection is.

4. Costs

4.1 Parts

The materials purchased were used either for the power subsystem or mechanical design. Most of the power subsystem items were used for controlling and distributing proper voltage throughout the circuit. The voltage regulators, motor driver, and even the ESP32-S3 microcontroller were used to regulate voltages for different subcomponents. The mechanical design was difficult due to the lack of experience with crafting materials. We have purchasedmany items that were intended to be used for the mechanical design, but putting everything together did not go as intended, ending up not using some of the materials we bought. For example, we have bought a bulk of ¼ rubber gasket rings to place on the mechanical design, but we ended up realizing they were not needed and used an alternate solution for the mechanical design instead.

Part	Manufacturer	Retail Cost ($)	Bulk Purchase Cost ($)	Actual Cost ($)
Printed Circuit Board	Remi	38.08	38.08	38.71
9V Alkaline Batteries	Amazon Basics	12.06	12.06	12.51
9V Buckle Connectors	California JOS	9.76	9.76	9.81
ESP32-S3	MakerHawk	47.98	47.98	48.49
5V Voltage Regulator	Niiyen	7.89	7.89	8.01
3.3V Voltage Regulator	STMicroelectronics	8.78	8.78	8.92
A4988 Motor Driver	WWZMDiB	14.99	14.99	15.23
Light Switch	Leviton	13.14	13.14	13.51
¼ Rubber Gasket Ring	Xilian Cui	6.49	6.49	6.89
Breadboard Jumper Wires	EDGELEC	6.98	6.98	7.21
Electronic Digital Measuring Tool	tengzhukeji	14.99	14.99	15.30
Total			143.06	145.88

Table 6. Cost of parts

4.2 Labor

Labor costs are calculated using the following formula:

Labor Costs = ideal salary (hourly rate) * actual hours spent

Where the ideal salary can be estimated at $16 per hour. Our distribution of work was concluded as Ruize Sun doing most of the work for the power subsystem, PCB design, and connecting everything through the WiFi subcomponent; Sangsun Lee doing the PCB soldering, managing plans and arrangements for the team, and communicating with TAs and professors; and Andrew Kim making the application and website for controlling the light switch.

Name	**Hours Spent**	**Labor Cost**
Ruize Sun	180	$2,880.00
Sangsun Lee	100	$1,600.00
Andrew Kim	60	$960.00
Total	340	$5,440.00

Table 7. Labor costs

Estimated Cost of Production: $5,585.88

5. Conclusion

5.1 Accomplishments

In conclusion, we have successfully implemented an automatic light switch that can be controlled by an application and website. Both have user interfaces that can be easily interacted with the ESP32-S3. The ESP32-S3 WiFi subcomponent was able to successfully connect to our application and website through HTTP server requests. Our ESP32-S3 converted voltages such that the motor driver can perform logic operations for the motor, and the motor would move under the instructions written at Arduino.

The project was able to move the light switch through our alternate mechanical design. Our mechanical design consisted of a pair of chopsticks that would move the switch up and down. We attached the chopsticks with double edged tape and glue sticks. The design works reliably without any problems throughout the testing process we have gone through, so our mechanical design was also successful.

In the end, we were able to control the motor successfully through arduino instructions, which were operated under the control of ESP32-S3. The application and website would send the instructions through the ESP32 WiFi module. The motor was able to move the chopsticks up and down, which would slide the switch up and down successfully. Through many attempts, we were able to reach our intended solution to control the light switch through user interface, so our project was successful.

5.2 Uncertainties

One of the uncertanties of our project includes the stability of our mechanical design. Although we have tested the mechanical design multiple times, we have not used the light switch as long to see when it fails to operate. Since we are using chopsticks to move the switch, there might be times where the chopsticks snap and the light switch fails.

Another uncertainty we have is the battery voltage that we can use to operate the light switch. The battery, when used for a long period of time, sometimes fails to meet the minimum voltage to operate the ESP32. This was one of the problems we faced during the demo, which was why the switch was not operating.

The last uncertainty we have is using a breadboard for circuitry instead of printed circuit board. Since we were not using a PCB, there might be some problems when our light switch is used in harsh environments or used for a long period of time.

5.3 Ethical considerations

One of the ethical issues to consider include placing this design in public places, because if only a few people have access to the light switch, then people might have issues using the switch. Another concern is that if anyone has access to the light switch application, people might use the application or website for malicious reasons. This might lead to wastage of electricity for people using the switch when not needed. Another problem to consider is that the light switch is only accessable through people with phones or computers, so people who do not have access to these devices would not have access to the light switch.

5.4 Future work

If we would further continue this project, we would attempt to improve the mechanical design such that it operates more stably. Changing the material of the mechanical design to a stronger material would make the light switch operate for a longer period of time. Additionally, we can add more features to the mechanical design so it can operate under multiple types of light switches, such as pole switches and rocker switches, to name a few. Currently, our design can only control the dimmer switch design, which limits the applicable usages in real life.

In the end, our team was able to learn how to creatively come up with a problem and solution using the ESP32-S3 microcontroller. Unfortunately, we were not able to meet one of our high-level requirements, which is the ability to control the light switch through voice control. This was mainly due to the lack of resources we could find online about the INMP441 microphone and how to connect it to the voice control module. If we continued this project, we would be focusing on implementing the voice control module for the light switch.

Further suggestions for work include adding a monitor to display the light switch status, turning the automatic light switch off when not used for a long period of time to save battery power, and reducing the overall design size to make the switch more compact and portable.

References

[1] Allegro Microsystems, “A4988: DMOS Microstepping Driver with Translator and Overcurrent Protection,” datasheet, Rev. April 5, 2022 Available: https://www.allegromicro.com/~/media/Files/Datasheets/A4988- Datasheet.ashx. Accessed Apr. 2, 2025.

[2] “Pololu - A4988 Stepper Motor Driver Carrier,” www.pololu.com. https://www.pololu.com/product/1182

[3] “PCB Trace Width Calculator | Trace Width Vs. Current Table,” mcl, Jun. 18, 2018. https://www.mclpcb.com/blog/pcb-trace-width-vs-current-table/

[4] INMP441 Omnidirectional Microphone with Bottom Port and I 2 S Digital Output GENERAL DESCRIPTION. https://invensense.tdk.com/wp content/uploads/2015/02/INMP441.pdf

[5] ESP32-S3-WROOM-1-ESP32-S3-WROOM-1U Datasheet 2.4 GHz Wi-Fi (802.11 B/G/N) and Bluetooth ® 5 (LE) Module Built around ESP32 S3 Series of SoCs, Xtensa ® Dual-Core 32-Bit LX7 Microprocessor Flash up to 16 MB, PSRAM up to 8 MB 36 GPIOs, Rich Set of Peripherals On-Board PCB Antenna.https://www.alldatasheet.com/datasheet pdf/pdf/1642038/ESPRESSIF/ESP32S3.html

[6] DPL12SV2424A21F3,DPL12S24PEncoders24DET21FRD/GRN, https://www.mouser.com/datasheet/2/418/3/NGDS11773449-0G-716223.pdf

[7] “Adafruit L9110H H-Bridge Motor Driver for DC Motors- 8 DIP.” 3DMakerWorld, Inc., 2025, 3dmakerworld.com/products/l9110h-h-bridge-motor driver-for-dc-motors-8-dip. Accessed 7 Mar. 2025.

[8] K. E. Elliott and C. M. Greene, “A local adaptive protocol,” Argonne National Laboratory, Argonne, IL, Tech. Rep. 916-1010-BB, 2006.

[9] IEEE. “IEEE Code of Ethics.” (2016), [Online]. Available: https://www.ieee.org/ about/corporate/governance/p7-8.html (visited on 02/08/2025).

Appendix A Requirement and Verification Table

Requirements	Verification	Verification status (Y or N)
The smart switch must maintain a stable and uninterrupted connection with the mobile app over a Wi-Fi range of at least 100 feet (30 meters) indoors, allowing users to conveniently control their lights from any room in the house, even through walls or obstacles.	Tested by sending control commands via the app to the smart switch placed 30 meters away indoors, with multiple walls and doors in between. Verified that the switch responded without delay or disconnection.	Y
The smart switch must execute app or voice commands within 2 seconds, providing near-instantaneous adjustments to the light level, ensuring a seamless and responsive user experience without noticeable delays.	Tested response time by sending app/web-page commands to the ESP32-S3 module and recording the delay between command issuance and motor activation. Results showed consistent execution within 0.5 seconds, well below the 2-second requirement.	Y
The voice module must accurately recognize and respond to spoken numbers between 1 and 3, adjusting the light brightness to the corresponding level with precise control, ensuring that users can easily set their desired light intensity via voice command.	Could not finish voice control module due to limited resources.	N

Table 8. High-level requirements and verifications

Requirements	Verification	Verification status (Y or N)
The MCU needs to be able to output a PWM signal to control the Motor Driver, and MCU should also know When should stop output PWM signal	Use an oscilloscope to measure the PWM output signal and the frequency is stable above 10kHz. And observe if the motor can stop when it reaches the desired angle. This information comes from the A/B phase signal of the incremental encoder.	Y
MCU needs to send a direction signal (DIR) through the GPIO port to control the forward and reverse rotation of the motor.	Connect the Motor Driver and observe whether the motor direction change matches the MCU control signal.	Y

Table 9. Requirements and verifications of board control subsystem

Requirements	Verification	Verification status (Y or N)
The motor shall support 5 distinct levels of movement, corresponding to switch positions	Commanded the motor to each of the 5 switch levels and measured displacement accuracy using a protractor	Y
The motor shall achieve an angular precision of ±20° to ensure accurate switch control	Compare actual motor rotation angle with the expected angle for multiple test cases. The average difference between the actual motor rotation angle and expected motor rotation angle should within ±20°	Y
The motor shall support bidirectional movement (up/down switch adjustment)	Verified bidirectional movement by setting the DIR pin high or low in C++ code and observing corresponding upward and downward motion of the motor.	Y

Table 10. Requirements and verifications of motor subsystem