ECE 445

Senior Design Laboratory - Fall 2023

Final Report

Distributed Light System

using Voice Recognition

Team 24

Anshul Goswami (anshulg3@illinois.edu)

Walter Tang (waltert3@illinois.edu)

Anish Naik (anishn3@illinois.edu)

TA: Kai Chieh (Jeff) Chang

December 6, 2023

Abstract

This final report will cover our Senior Design Project, the Distributed Lights System using Voice Activation. It covers our designs going into the project and the changes required to make it functional that we realized along the way. We set out to address the main issue of traditional light switches causing potential hazards around the house. Our project demonstrated that we can use voice recognition to help power the circuits that allow users to turn on/off lights around the house, potentially alleviating those hazards. This paper should contain enough information for someone to understand what we did and how to recreate it themselves.

1 Introduction.....1
1.1 Problem..... 1
1.2 Solution.....1
1.3 Block Diagram..... 2
2 Design.....3
2.1 Design Procedure.....3
2.1.1 Main Station Box.....3
2.1.2 Voice Recognition.....3
2.1.3 Lamp Box.....4
2.2 Design Details.....4
2.2.1 Main Station Box.....4
2.2.2 Voice Recognition.....4
2.2.3 Lamp Box.....4
3 Verification..... 6
3.1 High Level Requirements..... 6
3.2 Requirements and Verifications of Modules..... 7
4 Cost and Schedule..... 7
4.1 Cost..... 7
4.2 Requirements and Verifications of Modules..... 9
5 Conclusions..... 10
References..... 11
Appendix..... 12

1 Introduction

1.1 Problem

Traditional light switches can pose a significant problem when it comes to safety, particularly in the dark. Imagine you wake up in the middle of the night, and want to quickly grab a drink of water. While you stumble around for the light switch, you may inadvertently knock over objects, bump into furniture, or collide with walls, transforming a simple task into a potential hazard. This inconvenience isn't limited to late-night scenarios; during emergencies, the time taken to find a switch in the dark can become critical. Furthermore, if someone has a physical disability, needing to constantly access the switches can become a laborious task.

Even though smart switches represent a significant improvement in terms of convenience and control over traditional switches, a lot of the current smart switches available still have their problems. Some smart switches either require access to a physical remote or smartphone, or utilize technologies that cause a lot of privacy concerns, which many people do not find preferrable. People also just want to be comfortable, like if you are cozy in bed, you may not want to get up and walk all the way across the room to turn off your lights!

1.2 Solution

Our solution is to create a distributed voice activated light system which will let you control different lights around your house. We want to create a solution in which you can talk to a Main Station to turn lamps in different locations on and off. This way, one can easily access lights without needing to move around.

We want to create a Main Station Box containing a microphone and Raspberry Pi, which will pick up voices and use that in voice recognition processing. If the Main Station Box recognizes a command, it will then send the corresponding signals to the ‘Lamp Boxes’ which contain a microcontroller and a

circuit with a relay which will switch the connection to the outlet on or off, effectively turning a light on or off. We also want as much of the voice recognition to be done as locally as possible.

1.3 Block Diagram

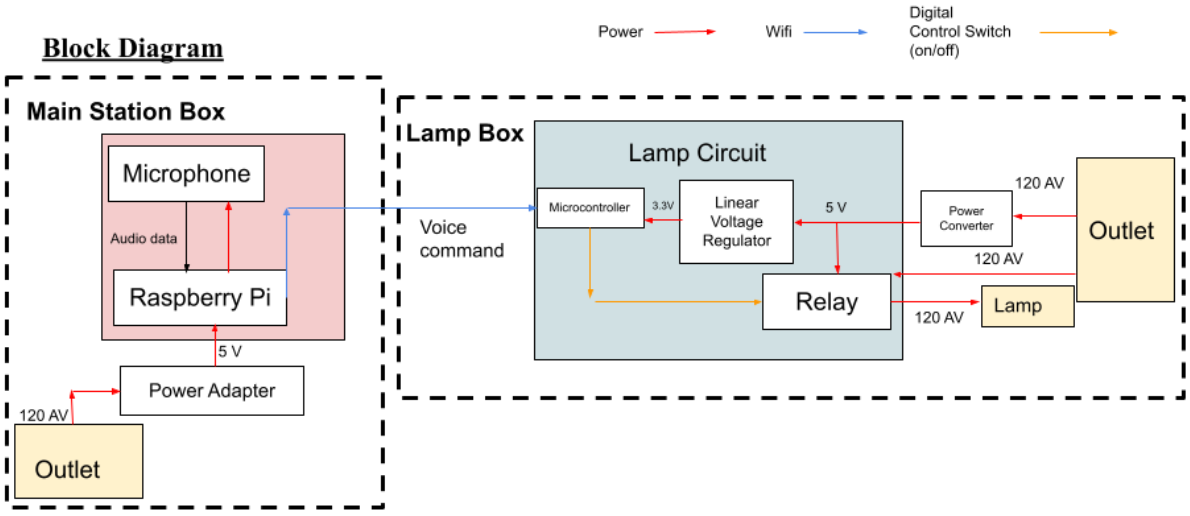

Figure 1: Block Diagram

Our System consists of two major components, the Main Station Box and the Lamp Boxes, which communicate using WiFi signals.

Our Main Station Box consists of a microphone, which picks up voices in the vicinity, and then sends that data to the Raspberry Pi that is connected to it. The Raspberry Pi is constantly running a Voice Recognition Module which converts the voice data into text. The Raspberry Pi is running a program that monitors for 6 specific commands, and if it detects one of them, it will send a command to the specific Lamp Boxes using Wi-Fi signals.

Our Lamp Boxes have a microcontroller that can receive the signal, which can then send a signal to the relay to switch its connection on or off. This relay connects an extension cable to an outlet, so when the Lamp Box's extension cable is plugged in an outlet, you can plug an actual lamp's power cable into the Lamp Box's outlet, and this controls the power connection between the actual outlet and the actual lamp.

So with these connections, our Main Station Box can pick up voice commands and then turn lamps on or off if they are connected to one of our lamp station boxes.

2 Design

2.1 Design Procedure

2.1.1 Main Station Box

For the Main Station Box, we needed two components: a Raspberry Pi to run the programs and a microphone. One of the high level requirements we had coming into designing our project was that we wanted our Main Station Box to pick up voice commands from up to a meter away. Because of this, we ended up using a much larger microphone that we initially planned. We also had the option for multiple kinds of Raspberry Pis with different memory sizes. Although we didn't put much thought into it when designing it, later we would run into an issue where the Raspberry Pi couldn't run the model we planned, so perhaps we could have accounted for that and got one with a larger amount of memory.

2.1.2 Voice Recognition

For voice recognition, we had a few other alternative approaches to our design. One design choice we were considering was deciding on which voice recognition library we wanted to use. While it would've been really cool to completely develop a voice recognition model from scratch and train our own words, that would take a lot of expertise and time and given our constraints we did not think that would be an approachable solution. We ended up having a few options to choose from, such as spchcat, AssemblyAi, and Vosk. We went with Vosk because Vosk is trained on the LibriSpeech dataset, which is a collection of 1000 hours of 16 kHz read English speech prepared by Vassil Panayotov and Daniel Povey, who are experts in speech processing, to reach an accuracy of 75%, which best fit our requirements. In addition, Vosk is an open-source free to use API, and has been used successfully in chatbots, smart home appliances, and virtual assistants similar to our project, so we determined that Vosk would be the best fit for our needs.

2.1.3 Lamp Box

Our design procedure involved many changes. Initially, we were going to use an AC 120V to 5V DC converter on our PCB. However, we decided not to do this. We instead went with providing the PCB 5V from a micro USB cable powered by a phone charger. This avoids us having to deal with Heat and additional sources of high current and voltage on our PCB. We definitely could have used a 120V AC to DC converter on our PCB, but the phone charger made it so that heat is generated outside of the PCB.

Another design choice we could have made was using a ESP8266 DEV board instead of an ESP-32E. This could have allowed us to not make a custom programming circuit, which is both expensive and takes a lot of space. However, PCBs usually don't have a dev board soldered on. They usually have SMD microcontrollers where the engineer designs their own programming circuit for their needs.

2.2 Design Details

2.2.1 Main Station Box

The Main Station Box uses a Raspberry Pi that runs a voice recognition program. It picks up voices using a microphone that is powered by and sends audio data using a USB serial connection. When the program recognizes one of the six commands, it sends a Wifi Signal to the Lamp Box on the LAN network using a 2.4 GHz network.

In order to make sure the microphone can pick up voices up to a meter away, we used eq. 1 to determine what decibels our microphone needs to be able to hear. Since the average human voice is about 60 decibels (dB), then a meter away it would be 20 dB.

$$\text{(eq. 1) } I(db)_2 = I(db)_1 - 20log(\frac{r_2}{r_1})$$

2.2.2 Voice Recognition

To utilize Vosk API, we loaded in our specified model using Python and integrated PyAudio with our microphone to discern sounds. This gave us our base model, but we still wanted to further improve the accuracy in case someone has an accent or doesn't speak as clearly. We implemented a simple feature

that checks if a similar sounding word is detected, for instance “outwit” instead of “outlet”, we can check the sequence of other words in the phrase to determine if a command was spoken or not. For example, “outwit one on" is most likely to be mistranslated as “outlet one on.” Implementing this feature improved our accuracy up to 87%.

2.2.3 Lamp Box

The Lamp Box had 3 main components. One was the ESP-32E chip itself. Second was the circuitry to program and power the microcontroller. Third was the circuitry to turn on and off the relay. The ESP-32E chip was soldered onto the PCB, and it was connected to the output of the voltage regulator in order to receive 3.3V. The PCB was also connected to RX/TX and boot pins, which allows our USB programer to put the microcontroller in boot mode and also transmit data via the RX/TX Lines.

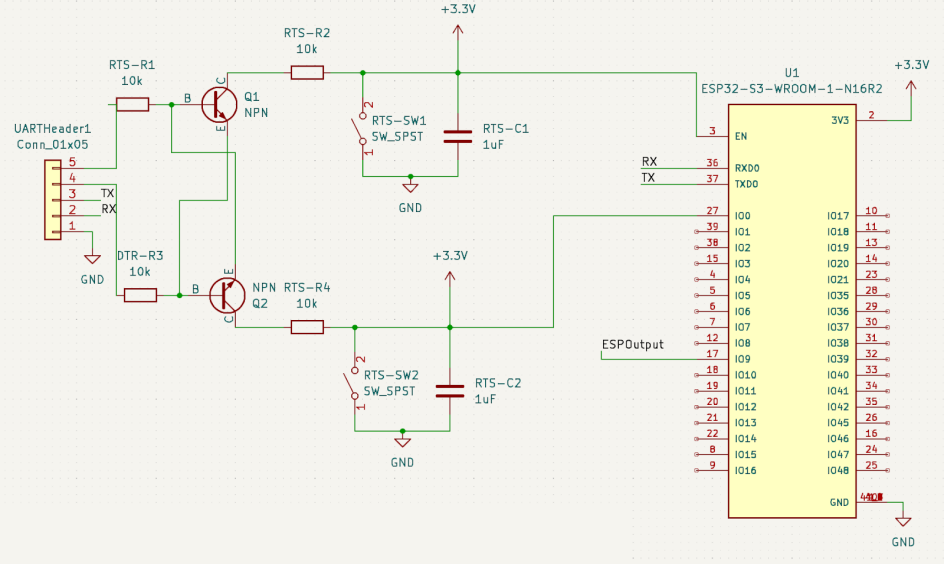

Figure 2: Programmer Circuit

We provide 5V via Micro USB but the ESP32-E needs 3.3V. Thus, we had a linear regulator circuit that took the 5V and converted it into 3.3V.

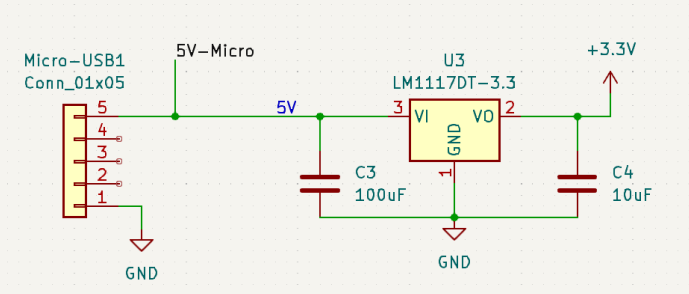

Figure 3: Linear Regulator Circuit

Our relay circuit consisted of a transistor and a flyback diode. The flyback diode prevented backwards EMF, since the relay had an inductor. The transistor allowed the relay to be powered by 5V. This 5V was directly provided by the Micro USB and not by the output of the voltage regulator. By sending a 3.3V signal to the transistor, it allowed the relay to be connected to 5V, thus turning it on. The transistor also allowed isolation between the ESP and the relay. Thus, if the relay sent too much power or something went wrong, the ESP-32's output pins would not short out.

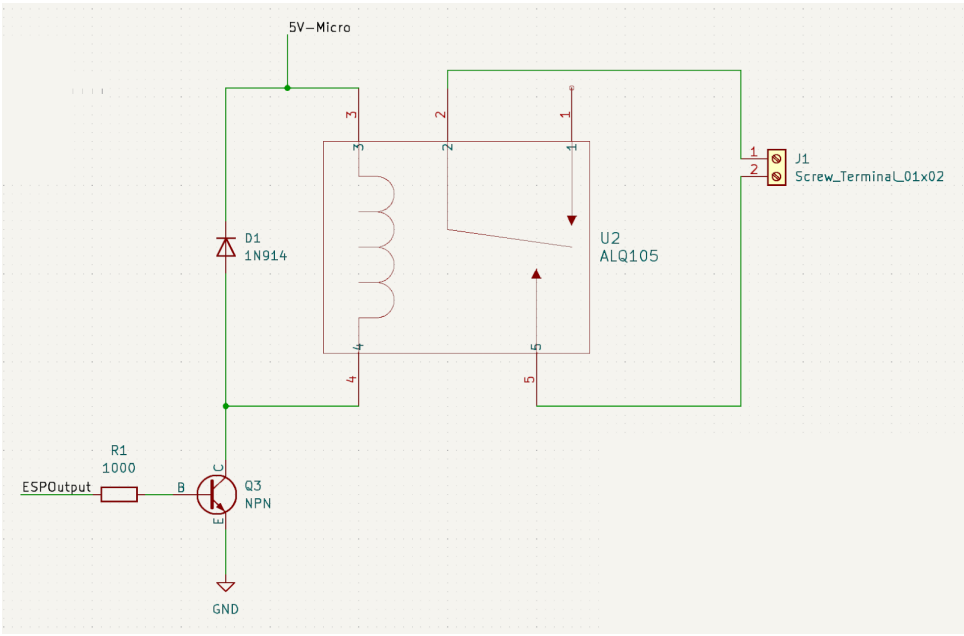

Figure 4: Relay Circuit

Our Lamp Box also consisted of an extension cord and an outlet socket. We stripped the non plug end of the extension cord and attached the corresponding wires to the outlet socket's screws. We then cut

the hot wire of the extension cord and attached the cut ends to both sides of the screw terminal in the relay circuit so that the relay could switch that connection on and off.

3 Verification

3.1 High Level Requirements

To consider our project successful, we had three high level requirements we wanted to fulfill:

The first one was that we wanted our system to be able to recognize 6 commands with 75% accuracy on our three different voices facing the microphone from a maximum of 1 meter from the Main Station Box. The system was able to recognize the 6 commands with 87% accuracy with three different voices, as shown by the tests on table 6 in the appendix. Our microphone was able to accurately pick up voices from up to a meter away, since we made sure to do the aforementioned tests in that range.

For our second high level requirement, we wanted to make sure our system could support up to two independent lamp boxes that can communicate with the main station box from a maximum of 5 meters away. We were unable to program the ESPs in our Lamp Box's circuits, so we were unable to test this. Instead we used an ESP development board, which we could program and was able to communicate with the Raspberry Pi since it could send HTTP GET requests to command the ESP chips, which were received even if it was almost 5 meters away from the Main Station Box. We only had one dev board to test it on, but since adding another dev board would add another IP in the same network, our design would have been able to support up to two Lamp Boxes that could communicate with the Main Box.

The third high level requirement we set was that we wanted visual feedback from the lamp boxes to be displayed within 5 seconds of the microphone picking up the command. As said before, we were unable to get the lamp box circuit programmed, so instead we had to use the dev board to test this. So using voice commands, we were able to turn an LED on and off. We timed it a couple times and it took about 2-3 seconds between receiving the voice command and lighting up.

3.2 Requirements and Verifications of Modules

The requirement and verification tables of the subsystems are in the Appendix. Our Main Station Box and our Voice Recognition Model fulfilled all the requirements and verifications we set out before starting the project, but our Lamp Box was unable to as we were unable to program it. We could confirm that the linear regulator was delivering the correct 3.3 V to the ESP with a multimeter. We were also unable to fulfill the requirement that the ESP should be able to switch the relay on and off since our ESP wasn't programmed, so we were unable to send a signal to do that. This meant that our lamp box circuit was unable to do what it needed since we couldn't use the circuit to turn lamps on and off using commands from the Raspberry Pi. We were easily able to fulfill the design requirement of the PCB being able to withstand 120V AC, since the PCB only received 5V after we made a design change. We could also plug in the extension cord and a lamp into the outlet socket, but since the relay was set to open we were unable to test if the coils and connections on our PCB could withstand the connection.

4 Cost and Schedule

4.1 Cost

Labor Costs (Per Partner)

($50/hr) * 2.5 * 12 hrs = $1,500 per week

(Total for 15 weeks: $22,500)

Table 1: List of Components and Costs

Part Name	**Quantity**	**Cost**
Raspberry Pi 4 Model B (2019)	1	$63.24
Raspberry Pi 4 Power Adapter	1	$9.99
Raspberry Pi 4 MicroSD Card	1	$9.97

Microphone for Raspberry Pi	1	$33.99
ESP-32E Microcontroller	2	$0 Via ECE Supply Store ($5.00 Retail)
Micro USB to pin Connector	2	$5.90
Linear Regulator	2	$3.61
Relay	2	$2.56
Screw Terminal	2	$2.50
Diode	2	$1.82
NPN Transistor	8	$3.36
Switch	4	$5.08
Outlet Socket	2	$0 Via Machine Shop ($9.98 Retail)
Extension Cables	2	$0 Via Machine Shop ($7.98 Retail)
Resistors	10	$0 Via ECE Supply Store ($1.00 Retail)
Capacitors	8	$0 Via ECE Supply Store ($0.80 Retail)
Total Cost		**$142.02**

4.2 Schedule

Table 2: Project Schedule

Week	Task	Person
September 24 - 30	Order Raspberry Pi	Walter Tang

	Start Designing PCB	Anshul Goswami
October 1st - 7th	Design Review	Everyone
	PCB Review	Everyone
	Speak with Machine Shop	Anish Naik
October 8th - 14th	First Round PCB Orders	Walter Tang
	Teamwork Evaluation	Everyone
	Final Machine Shop Revisions	Everyone
	PCB Revisions	Anish Naik
	Start working with Raspberry Pi	Anshul Goswami
October 15th - 21st	Second Round PCB Orders	Anish Naik
	PCB Revisions	Anshul Goswami
	Get Parts	Everyone
	Start Testing Speech Recognition API	Walter Tang
October 22nd - 28th	Pass Audit	Everyone
	Individual Team Progress Report	Everyone
	Third Round PCB Order	Anshul Goswami
	Start Assembling	Anish Naik
	Work with Microcontrollers	Walter Tang
October 29th - November 4th	PCB Revisions	Anshul Goswami
	Continue Assembling	Anish Naik
	Start Testing	Everyone
November 5th - 11th	Continue Testing	Everyone
November 12th - 18th	Mock Demo	Everyone
	Fix Bugs	Everyone
November 19th - 25th	Fall Break	N/A
November 26th - December 2nd	Final Demo	Everyone

December 3rd - 9th	Final Presentation	Everyone

5 Conclusions

In conclusion, our final PCB was not 100% functional. Although we were able to verify that some of our modules were in fact working, in the end, the ESP8266 was not functioning, thus we were not able to turn on our relay, which controls our lamp.

What was working

- Communication between our Raspberry Pi and ESP-32E. The ESP32-E successfully hosted a web server and the raspberry pi was able to send GET requests for different commands. We verified this by using an ESP8266 Dev Board instead of our PCB, and made the DevBoard flash the in-built LED to acknowledge the recipient of different commands
- Our Voltage regulator circuit was working. We confirmed this was working by looking at the output that the ESP32-E received using a multimeter.

What was not working

- Our ESP32-E chip did not send back serial data. Thus, it was not turning on at all.
- Due to the ESP-Chip not working, we were not able to test if the relay was turning on properly

Possible expansions of our project

While we got most of our high level requirements working, we feel that there could be a lot of possible expansions for our project. One feature we would like to add is being able to support more commands. Turning a light on/off is very standard, and we would like to support more commands such as setting timers for lights and changing light colors. Another feature we would like to add is to be able to support lamp boxes to control more lights around the house. This way, our project will become a lot more scalable and easy to use, becoming an all in one product to manage all light switches in the house, potentially making it a possible contender in the market for smart switches.

Ethics and Safety

In terms of ethics, we followed the IEEE Code of Ethics [6]. As engineers, we know that technologies have the ability to affect someone's life and we hold ourselves to the highest ethical standard when working professionally including but not limited to:

1. **Protect the safety and privacy of the public** [6]

   IEEE Code of Ethics states "to respect and protect the privacy of individuals." Our voice recognition system will not store any data offsite to ensure data privacy of individuals.

2. **Ensure that we treat each team member with respect and ensure that each person on the team follows the code** [6]

   In order to establish good communication between the team and TA, we established a Discord server to communicate meetings and deadlines. With our GitLab repository, we store a journal detailing each member's work to not only keep track of technical details but also attributing contributions correctly.

3. **Consistently learning and improving our abilities during the process** [6]

   The purpose of this class is to give us an opportunity to demonstrate the skills we have learned and apply them to a real world application. We will follow all required protocols, such as the CAD training, as well as consult our TAs and professors to ensure that requirements and deadlines are being met. We will also consistently ask for feedback to ensure that we are doing the best that we can.

In regards to safety regulations:

1. We will ensure that our project follows any relevant license terms of service for all software used.
2. When building any physical materials needed for our project, we will follow the required safety guidelines in the OpenLab to ensure the safety of everyone involved.
3. Since we will be working with AC power, we will ensure that each member completes the High Voltage training and adhere to these guidelines:

a. Wearing proper safety gear - we will wear protective gear such as safety goggles and insulating gloves when working with the circuit connected to AC 120V.
b. Before making any adjustments to the circuit, we will ensure that it is completely isolated from the power supply.

References

[1] Asma Trabelsi, Sébastien Warichet, Yassine Aajaoun, Séverine Soussilane,
Evaluation of the efficiency of state-of-the-art Speech Recognition engines,
Procedia Computer Science, Volume 207, 2022, Pages 2242-2252, ISSN 1877-0509,
https://doi.org/10.1016/j.procs.2022.09.534.
(https://www.sciencedirect.com/science/article/pii/S1877050922014338)

[2] P. Setiawan and R. Yusuf, "IoT Device Control with Offline Automatic Speech Recognition on Edge Device," 2022 12th International Conference on System Engineering and Technology (ICSET), Bandung, Indonesia, 2022, pp. 111-115, doi: 10.1109/ICSET57543.2022.10010962.
https://ieeexplore.ieee.org/document/10010962

[3] Panayotov, Vassil, et al. *LIBRISPEECH: AN ASR CORPUS BASED on PUBLIC DOMAIN AUDIO BOOKS*. https://www.danielpovey.com/files/2015_icassp_librispeech.pdf

[4] "VOSK Offline Speech Recognition API," *VOSK Offline Speech Recognition API*.
https://alphacephei.com/vosk/

[5] "ESP32-S3-WROOM-1 Datasheet v1.2." Espressif Systems, 2023
https://www.espressif.com/sites/default/files/documentation/esp32-s3-wroom-1_wroom-1u_datasheet_en.pdf

[6] IEEE. ""IEEE Code of Ethics"." (2016), [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html (visited on 02/08/2020).

Appendix

Table 3: Main Station Box - Requirements & Verification

Requirement	Verification
The Raspberry Pi will be able to power and communicate the microphone via the USB Protocol.	● Plug the microphone into the Raspberry Pi ● Ensure the light on microphone that shows it is powered is on ● Check that the microphone is detected by the Raspberry Pi
There will be a cooldown of 4 ± 1 seconds between each successful voice command recognized.	● Monitor the text data in Raspberry Pi that was converted from the audio data gathered ● Repeatedly say a command at 1 second intervals ● Make sure that it is processing every 5th time the command is set
Our microphone will detect voices between 20dB to 60dB, which is the average sound intensity of speaking up to one meter away.	● Monitor the text data in Raspberry Pi that was converted from the audio data gathered ● Say a command while 1m away from the microphone ● Repeat Previous Step at .5m away and .1m away ● Ensure the Raspberry Pi picked up the 3 commands
Send commands to ESP32-E using Bluetooth signals on the LAN network using 2.4 GHZ network	● Send a Dummy Signal to ESP32-E 3 times from Raspberry Pi ● Make sure the ESP32-E receives the command 3 times using the programmer

Table 4: Voice Recognition- Requirements & Verification

Requirement	Verification
Perform voice recognition using Raspberry Pi and the Vosk API with 75% accuracy.	● Test the voice recognition model using the same consistent commands using the same voice (“Outlet 1 On”, “Outlet 1 Off”, “Outlet 2 On”, “Outlet 2 Off”, “All Outlets On”, and “All Outlets Off”) ● Ensure that each command will be recognized at least 3 out of 4 times

Can perform voice recognition of the six commands (listed in high-level requirements) using our three different voices.	• Test the voice recognition model with three different voices and the same consistent commands (“Outlet 1 On”, “Outlet 1 Off”, “Outlet 2 On”, “Outlet 2 Off”, “All Outlets On”, and “All Outlets Off”) • Ensures that each command will be recognized at least 3 out of 4 times

Table 5: Lamp Box Subsystem - Requirements & Verification

Requirement	Verification
Our Linear Regulator circuit will take 5V input DC and convert it into a 3.3V DC in order to power the ESP32-E	• Using a Multimeter with one end connected to ground, measure the input and the output voltages. • Verify the input is 5V and the output is 3.3V
Receive signals from Raspberry Pi to turn Relay on/off	• Send a signal to turn the relay on and off with the Raspberry Pi • Hear a clicking sound for confirmation
The PCB should be able to withstand 120V AC without the PCB generating too much resistance, thus generating heat	• Test to see the resistance of the 120V power lines in PCB can withstand the load of 120V Mains by measuring its resistivity with a multimeter • Use a thermistor in order to measure the heat on the PCB and make sure that the temperature is not too high to disrupt circuit function

Table 6: Voice Recognition Accuracy

	Anish 1	Anish 2	Anish 3	Anshul 1	Anshul 2	Anshul 3	Walter 1	Walter 2	Walter 3
outlet one on	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	TRUE
outlet one off	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
outlet two on	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE

outlet two off	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
all outlets on	FALSE	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE
all outlets off	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE