University of Ottawa
Faculty of Engineering
Department of Engineering

GNG2101 - Introduction to Product Development & Management For Engineers

Prof. Mana Azarm

TA Jack Walsh

PM Kyla

PM Adi

Project Deliverable I – User Manual

C13 - Talk Box

Student Name	**Student Number**
Zainab Badawi	300034146
Tia El Masry	300160596
Kian Mozafarian	300138481

April 11, 2021

Table of Contents

List of Figures

List of Tables

List of Acronyms and Glossary

Table 1. Acronyms

Acronym	**Definition**
GUI	Graphical User Interface
PC	Personal Computer
PHP	Hypertext Preprocessor

1 Introduction

This user manual is a detailed breakdown of the project codenamed: TalkBox. As a group we decided to name our final design and solution SuperSpeak, thus TalkBox will be referred to as SuperSpeak for the rest of this manual. In this project, we decided to make a device that helps people with physical impairments that have cognitive disabilities and have difficulty speaking. In this user manual, we will be going over things to look out for, tools to use for development, device capabilities, and product troubleshooting & development.

2 Overview

There are over 440,000 Canadians who have significant speech, language and communication disabilities that limit their ability to do the things we tend to do on a normal day to day basis. Without proper communication, one can expect themselves to be heavily weighed down when trying to talk to family and friends, or even ask for help. The matter grows more concerning, A Canadian study shows that Almost, 1 million Canadians adults were identified as having a dexterity disability and more than 69% of them have reported having some level of unmet need for everyday activity. With the growing senior population these limitations are way too common.

The device is meant to help the disabled community that suffer from cognitive speech and dexterity complete simple and customizable daily tasks independently. The team worked with the client to extract the most fundamental needs that the SuperSpeaker was produced to satisfy. Firstly, the user should be able to control the communication commands through a smart button or joystick. The controller (joystick) should have a large surface area and requires a low force to operate. Secondly, the commands of the SuperSpeaker can produce sounds of simple phrases and can automate simple daily tasks. The features and phrases of the SuperSpeaker can be customized by a separate user interface. Finally, the device needs to be modular and upgradable down the line.

Our team has carefully examined our client needs and created a device that provides a 3.5 inch screen that demonstrates phrases and icons for the user. A joystick that allows the user to select phrases organized in a database with a very low required force in the x-axis relative to the hand motion. and A simple user interface that can be accessed from any device (desktop, laptop, smartphone) to edit the interface.

Other organizations that produce similar products to ours are Tecla, who developed a smart button called Tecla-e to help people with disabilities do home environmental tasks through the use of the tecla app. Another is, Technology for independent living, provides custom smart devices to aid the use of home automation appliances for disable communities in BC.

Even though our product is more affordable than existing products, Super Speaker is open source and is adaptable for future upgrades Unlike our competitions, interaction with our device is not limited to a phone application or a webpage, our solution offers a user interface that is designed to display icons and phrases for the user to choose from. Additionally, a webpage that can be accessed from any device from anywhere in the world anytime to update the phrases in the Super Speaker.

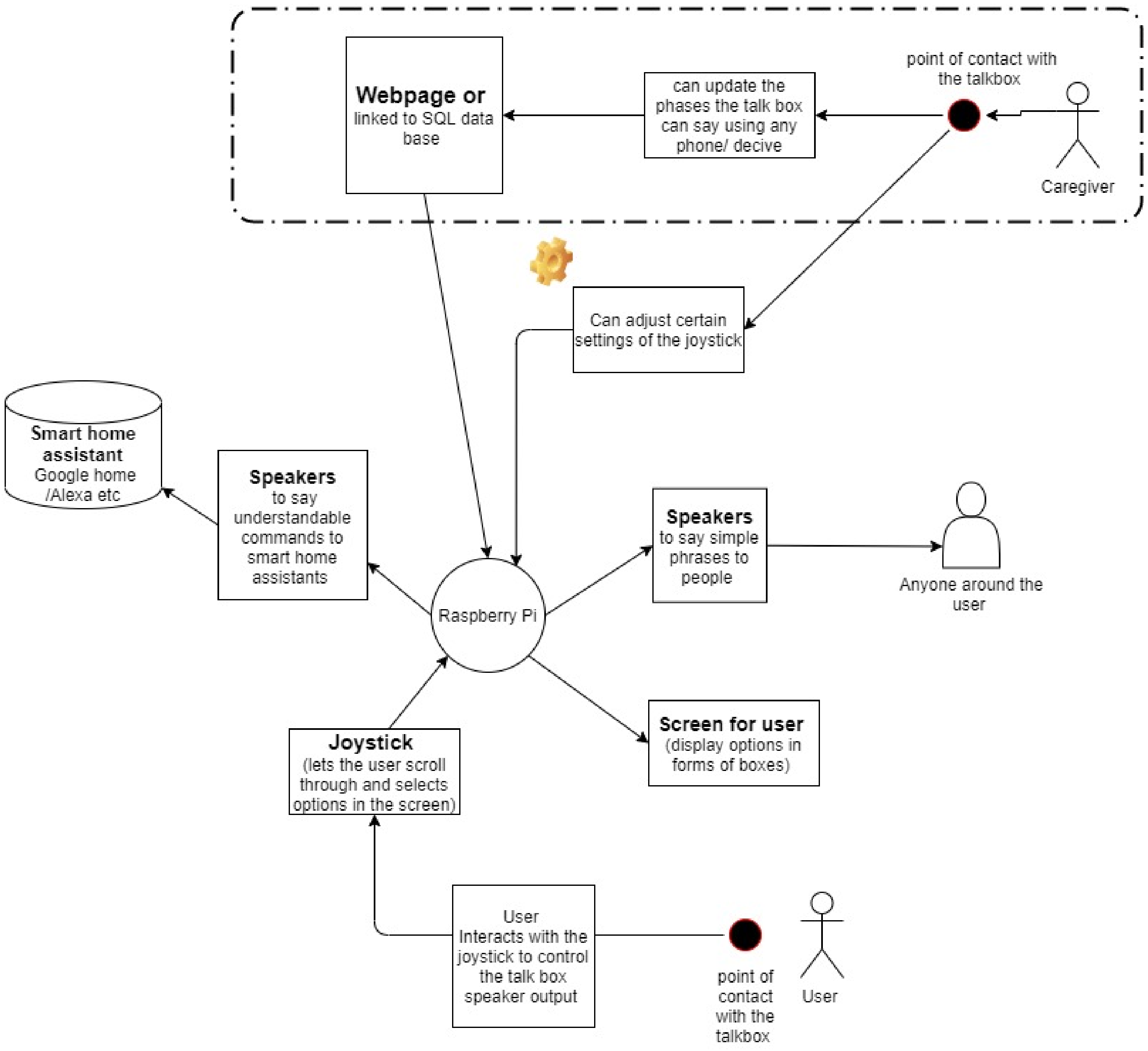

Webpage or
linked to SQL data base
can update the phases the talk box can say using any phone/ decive
point of contact with the talkbox
Caregiver
Can adjust certain settings of the joystick
Smart home assistant
Google home /Alexa etc
Speakers
to say understandable commands to smart home assistants
Raspberry Pi
Speakers
to say simple phrases to people
Anyone around the user
Screen for user
(display options in forms of boxes)
Joystick
(lets the user scroll through and selects options in the screen)
User
Interacts with the joystick to control the talk box speaker output
point of contact with the talkbox
User

Figure 2-1:Talk Box Functionality Breakdown

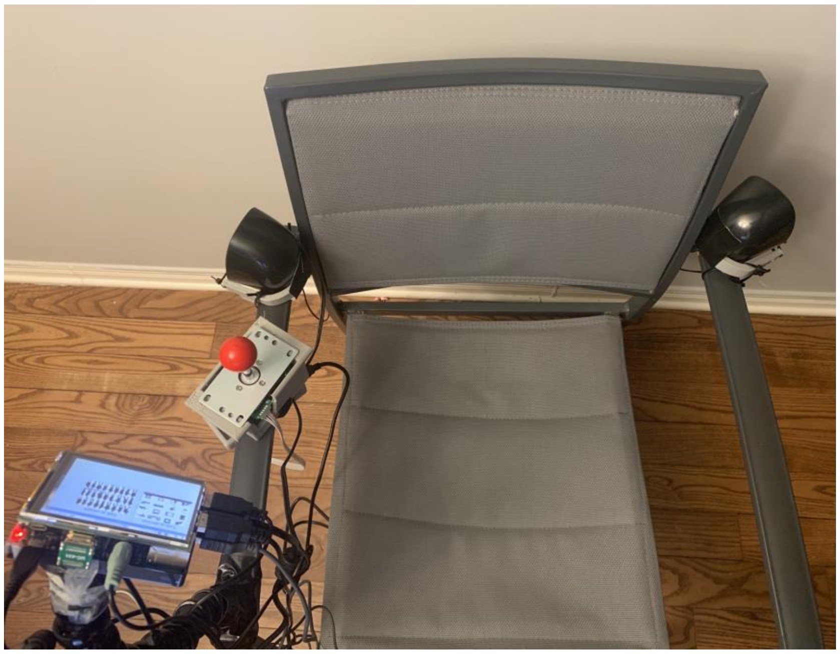


Figure 2-2: Gooseneck with user screen, joystick and speakers mounted on chair

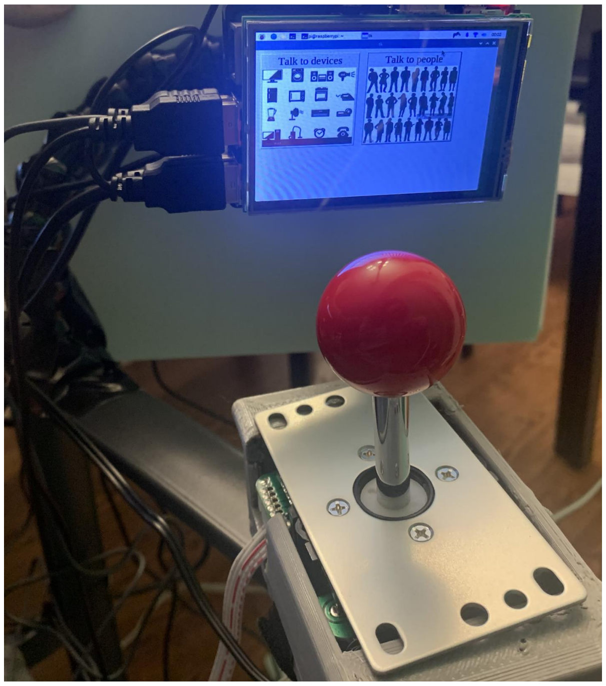


Figure 2-3: user main display panel

This Prototype objective is to establish functionalities that allow the raspberry Pi to output sound, and display options for the user to pick between speaking with devices or people (shown in **Figure2-3**: user main display panel). In addition is to allow the user to navigate between options by moving the joystick in the x-axis then selecting the option by pushing on the joystick downwards. Finally, the caregiver is able to add phrases to the sql data phase through a webpage that is hosted in one of the group members existing webpage server. (**Figure2-1**:Talk Box Functionality Breakdown) demonstrates a detailed breakdown of the talk box functionalities.

2.1 Cautions & Warnings

Even though the wires and all electronics are sealed, the user should avoid spraying or dipping the device in any liquids, as it could damage the computer on board. For cleaning purposes, the user should use a damped cloth to safely wipe down the surfaces of the device. If the raspberry pi detects a problem, it may demonstrate one of the following symbols:

<u>Undervoltage warning</u>

If the power supply to the Raspberry Pi drops below 4.63V (+/-5%), the following icon is displayed.

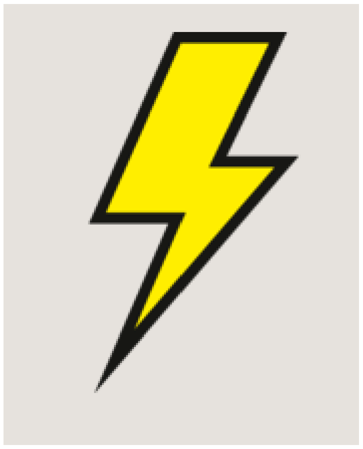


Figure 2-4: Raspberry pi Undervoltage warning

<u>Over temperature warning (80℃-85℃)</u>

If the temperature of the SoC is between 80°C and 85°C, the following icon is displayed. The ARM core(s) will be throttled back in an attempt to reduce the core temperature.

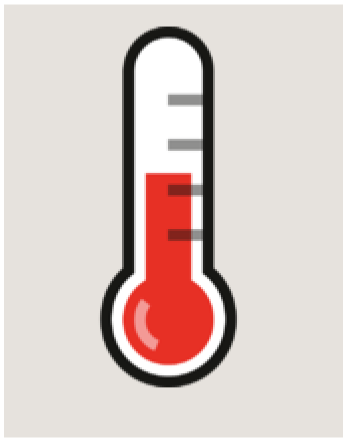


Figure 2-5: Raspberry pi Over temperature warning (80°C-85°C)

<u>Over temperature warning (over 85°C)</u>

If the temperature of the SoC is over 85°C, the following icon is displayed. The ARM core(s) and the GPU will be throttled back in an attempt to reduce the core temperature.

Figure 2-6: Raspberry pi Over temperature warning (over 85°C)

3 Getting started

In this section, we will give a simplified example of how to properly use SuperSpeak, maintain any changes, and how to safely navigate and exit the software/ turn off the computer.

3.1 Set-up Considerations

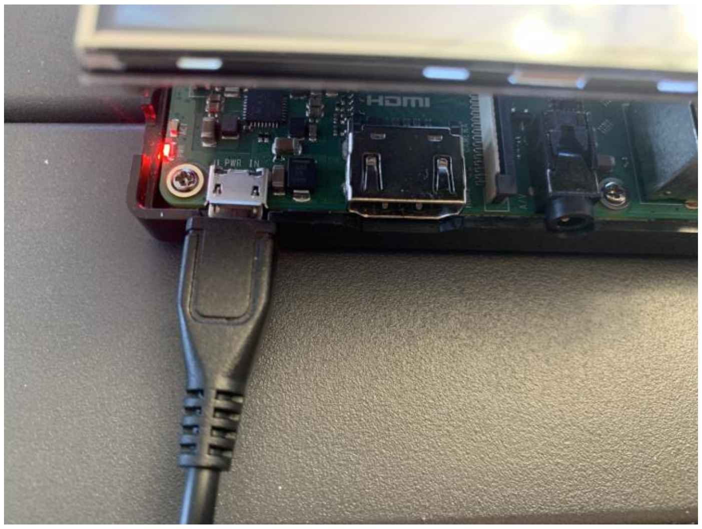


Figure 3-1: Power Cable

For starters, the device needs to be powered. The Raspberry Pi computer is powered via a USB type A, and it needs to be connected to a wheelchair battery via XLR cable on the providing end. The cables can be managed via reusable zip ties or special mounts for cable management.

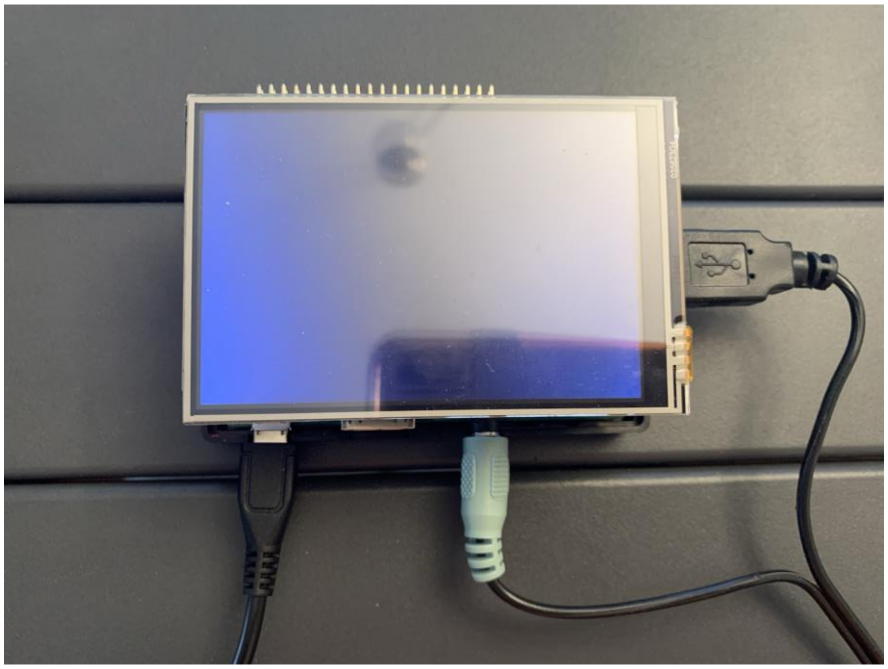


Figure 3-2: Cable Connection

After the device is powered, it can be mounted on the gooseneck mount made specifically for the Pi and then attached to the wheelchair. From there, the appropriate position can be adjusted for the user. When the device is turned on, it will have the SuperSpeak program ready and running; the main task the user which will be setting up the device has to do is turn on the wifi, alike how it is demonstrated in the pictures below.

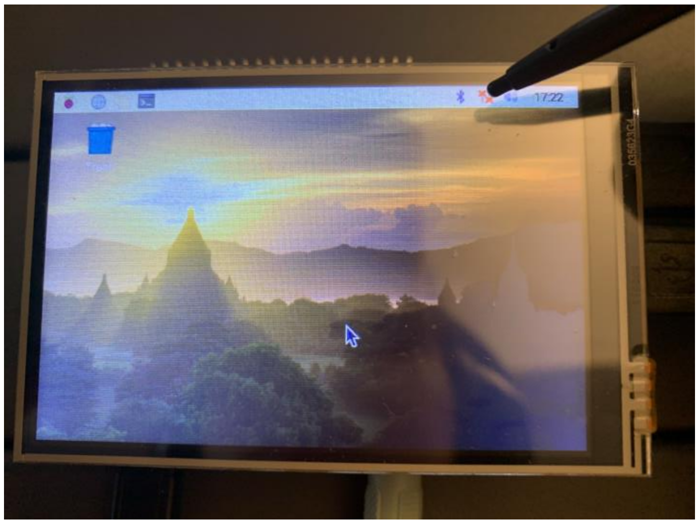


Figure 3-3: Icon showing device is disconnected from network

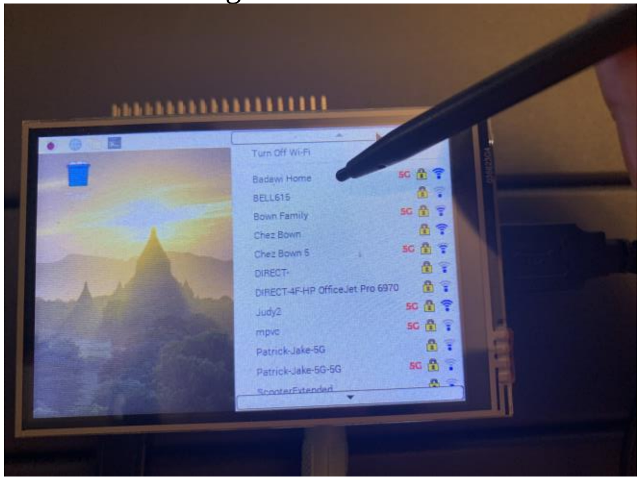


Figure 3-4: Select the proper network you want to connect to

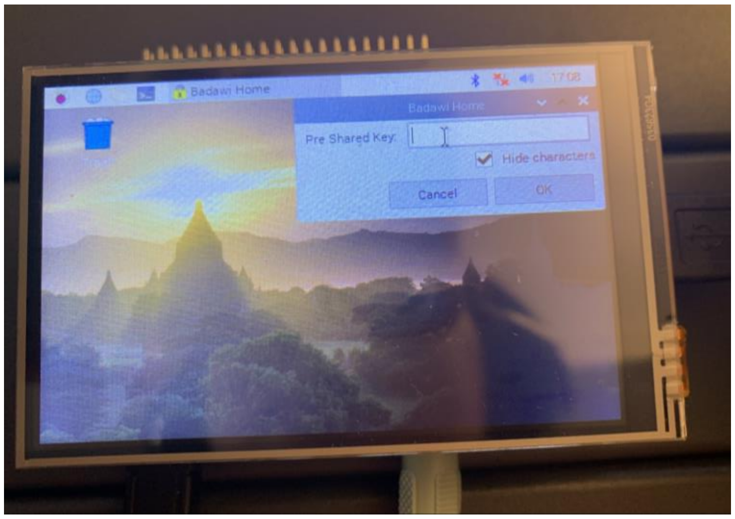


Figure 3-5: Inputting the wifi password

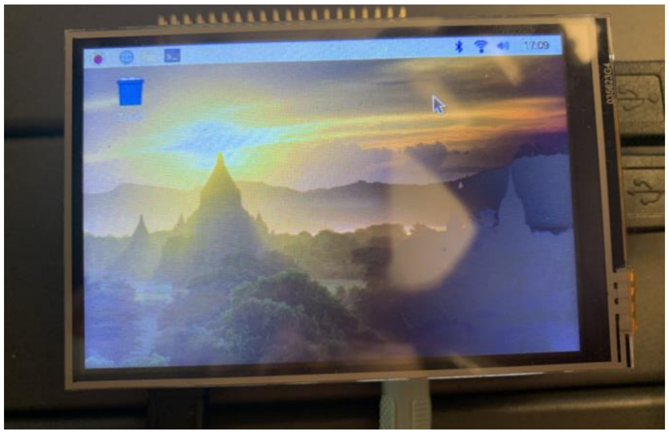


Figure 3-6:Wifi indicator on top right of screen should show connection

The joystick additionally the joystick antiMicro confirmation device needs to be activated by simply clicking on it on the touch screen or with a mouse. The antiMicro has the icon displayed in figure 3.1 and can be found on the home screen as soon as you boot the device.

Figure 3-7: Joystick Configuration Program

3.2 User Access Considerations

- Caretaker:
    - Can update the phrase list that is being used by the SuperSpeak app via the web service provided
    - Can set-up the Pi and connect it to a network and navigate the device via touch screen
    - Restrictions include anything regarding coding or hardware maintenance

- User In Wheelchair:
    - Can navigate through the SuperSpeak app via the joystick only

3.3 Exiting the System

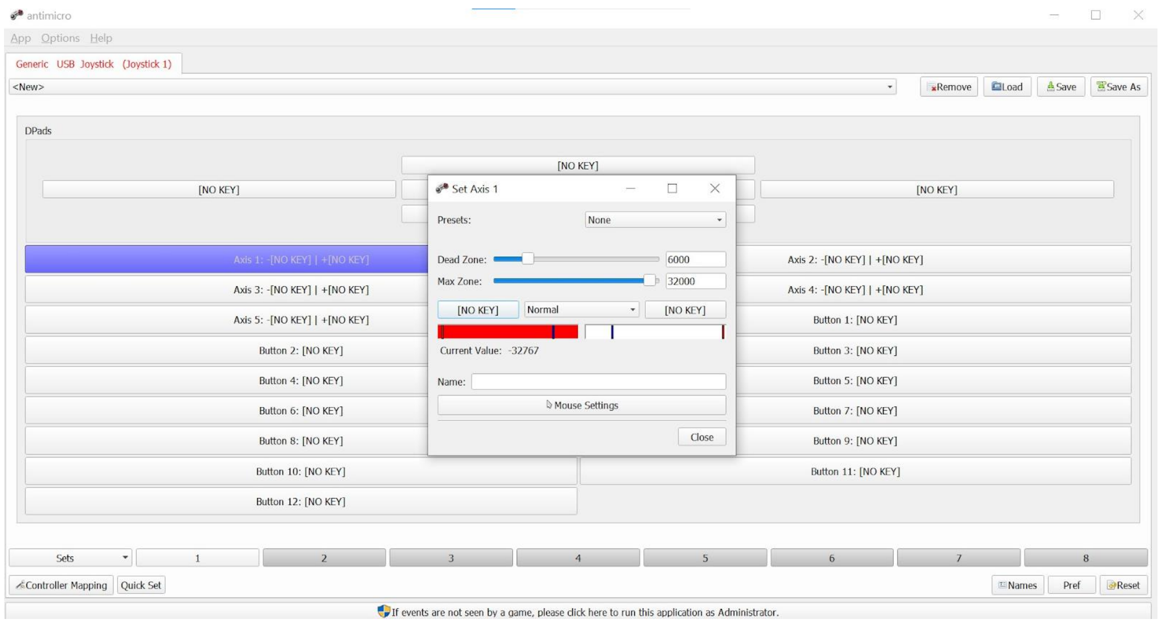
The system is meant to be set up and stay up and running until technical maintenance needs to be performed or the device needs to be removed. In either case, in order to shut down the system safely, a person with enough dexterity needs to close the python application by hitting the ‘x’ button on the top right of the screen; then shutdown the Pi from the operating system.

4 Using the System

The use of Superspeak is simple. Upon booting the device the python script that runs, the main program automatically runs.

4.1 Joystick Settings

If the user would like to change the default configuration of the joystick they can click on the antiMicro application found in the raspberry pi home screen where they can move the joystick left, right, up, and down and change the corresponding keyboard key associated with it. Figure 4.2 to Figure 4.5 demonstrate a step by step method of updating the joystick configuration.



Presets: None

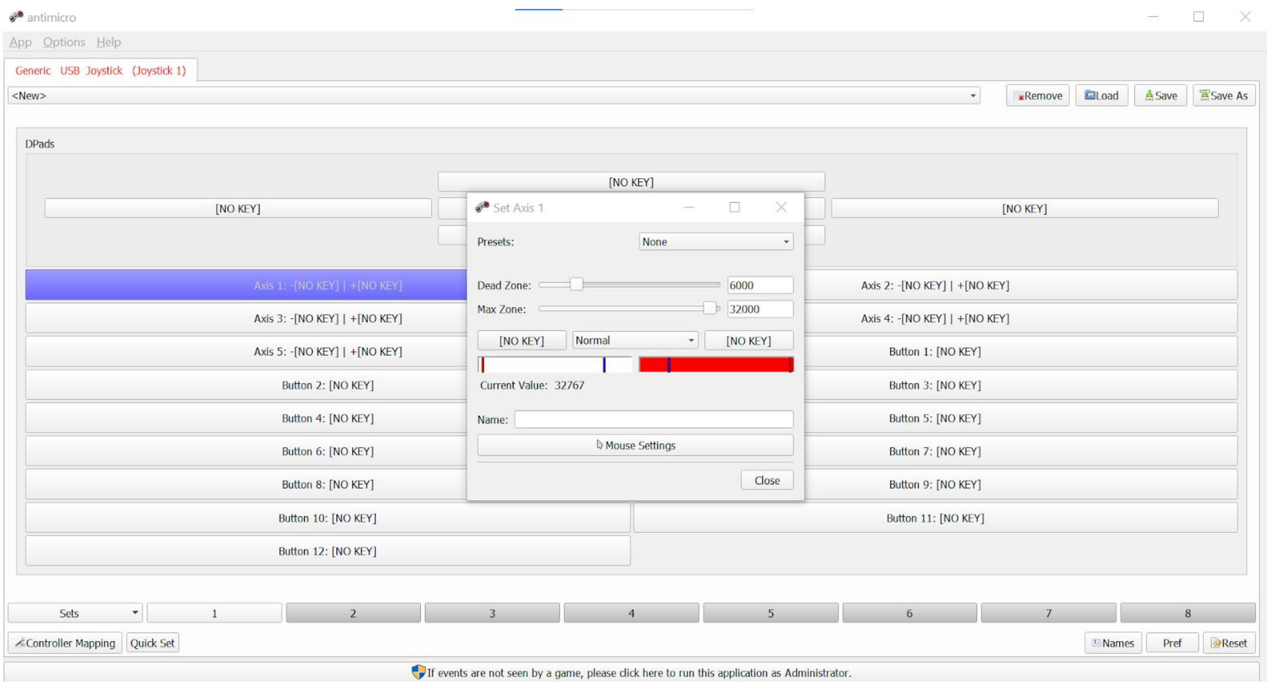
Dead Zone: 6000

Max Zone: 32000

Current Value: -32767

Name:

Figure 4-1: Moving the joystick to the left



Presets: None

Dead Zone: 6000

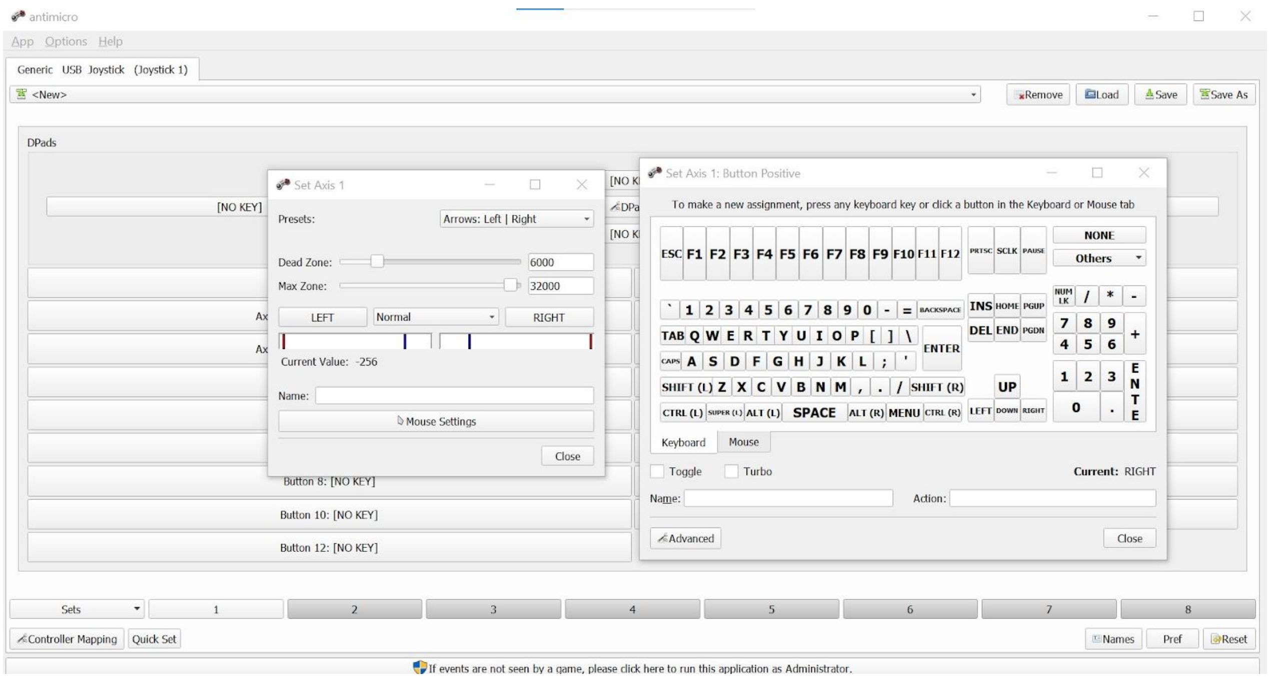
Max Zone: 32000

[NO KEY] Normal [NO KEY]

Current Value: 32767

Name:

Figure 4-2: Moving the joystick to the right



Presets: Arrows: Left | Right

Dead Zone: 6000

Max Zone: 32000

LEFT Normal RIGHT

Current Value: -256

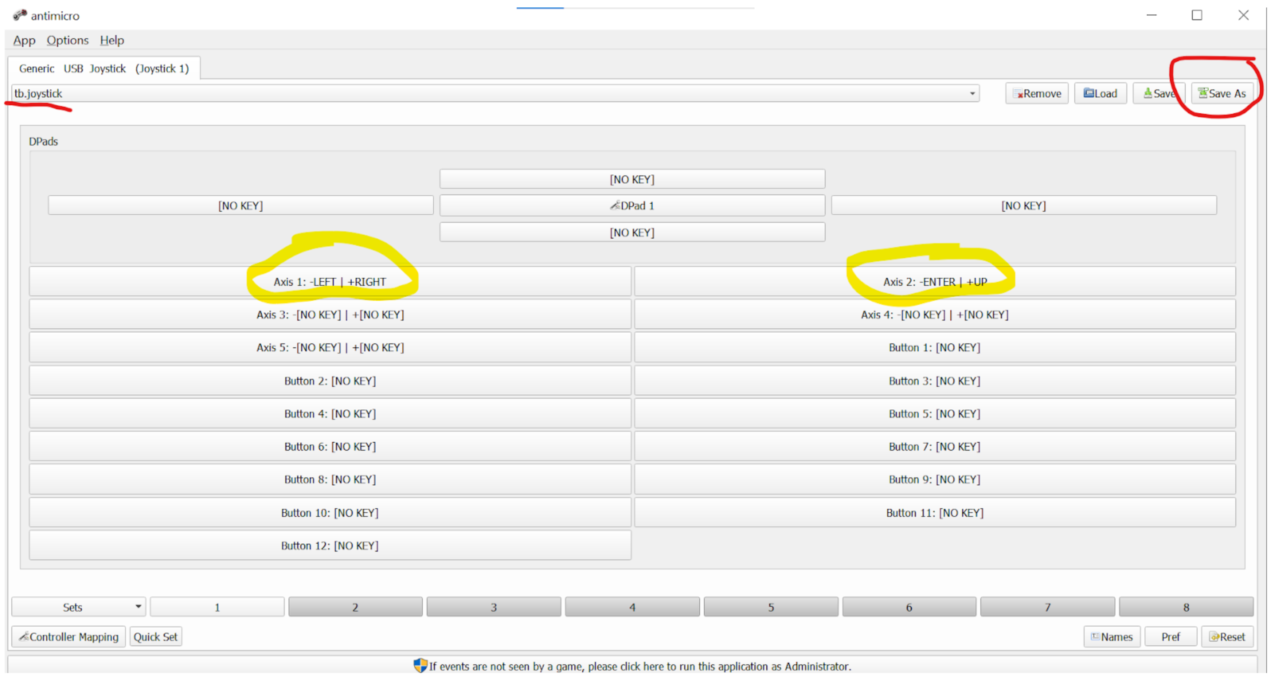
Name:

To make a new assignment, press any keyboard key or click a button in the Keyboard or Mouse tab

Toggle Turbo Current: RIGHT

Name: Action:

Figure 4-3: adding a keyboard key to correspond to joystick movement


antimicro
App Options Help
Generic USB Joystick (Joystick 1)
Remove
Load
Save
Save As
DPads
DPad 1
Sets
1
2
3
4
5
6
7
8
Controller Mapping
Quick Set
Names
Pref
Reset
If events are not seen by a game, please click here to run this application as Administrator.

tb.joystick

[NO KEY]
[NO KEY] DPad 1 [NO KEY]
[NO KEY]

Axis 1: -LEFT | +RIGHT
Axis 2: -ENTER | +UP
Axis 3: -[NO KEY] | +[NO KEY]
Axis 4: -[NO KEY] | +[NO KEY]
Axis 5: -[NO KEY] | +[NO KEY]
Button 1: [NO KEY]
Button 2: [NO KEY]
Button 3: [NO KEY]
Button 4: [NO KEY]
Button 5: [NO KEY]
Button 6: [NO KEY]
Button 7: [NO KEY]
Button 8: [NO KEY]
Button 9: [NO KEY]
Button 10: [NO KEY]
Button 11: [NO KEY]
Button 12: [NO KEY]

Figure 4-4: Saving the file to update the changes to the joystick

4.2 Navigating the main system

The user can use the joystick default configuration to scroll (left and right) between the different categories found in the home page: Talk to devices and Talk to people, as shown in figure 4.5: SuperSpeak home page.

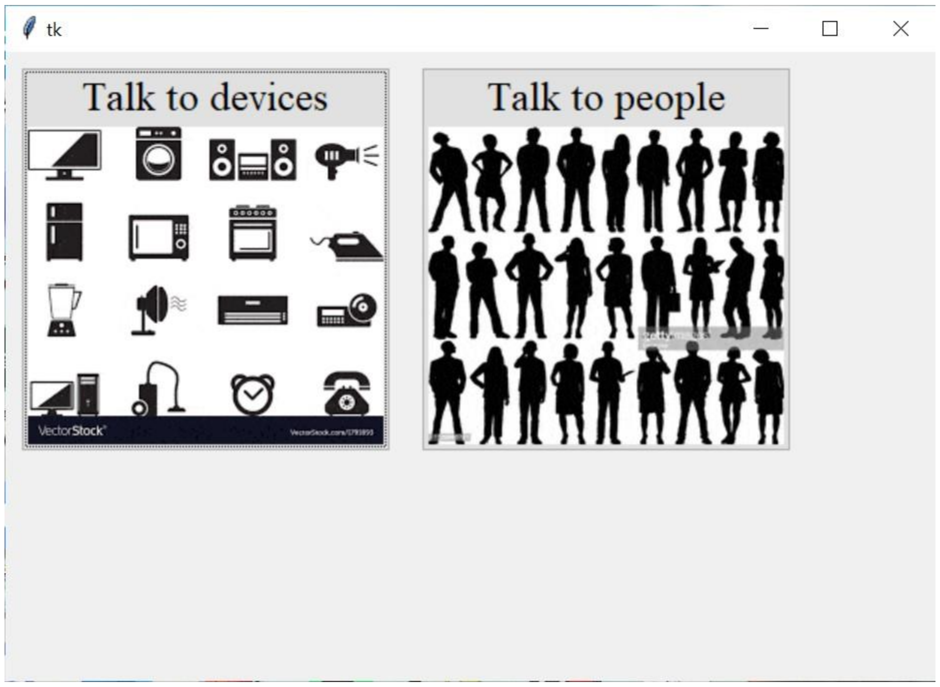


Figure 4-5: SuperSpeak home page

Once the user selects a category, they can scroll though the phrases in each category and then move the joystick upwards to select the phrase that they wish to be outputted. The phrase the user lands on is highlighted as shown in figure 4.6: Highlighted button in the talk to people category. As the user scrolls further to the left or right beyond the buttons that can fit in the screen, the program automatically scrolls to display the buttons the user is selecting.

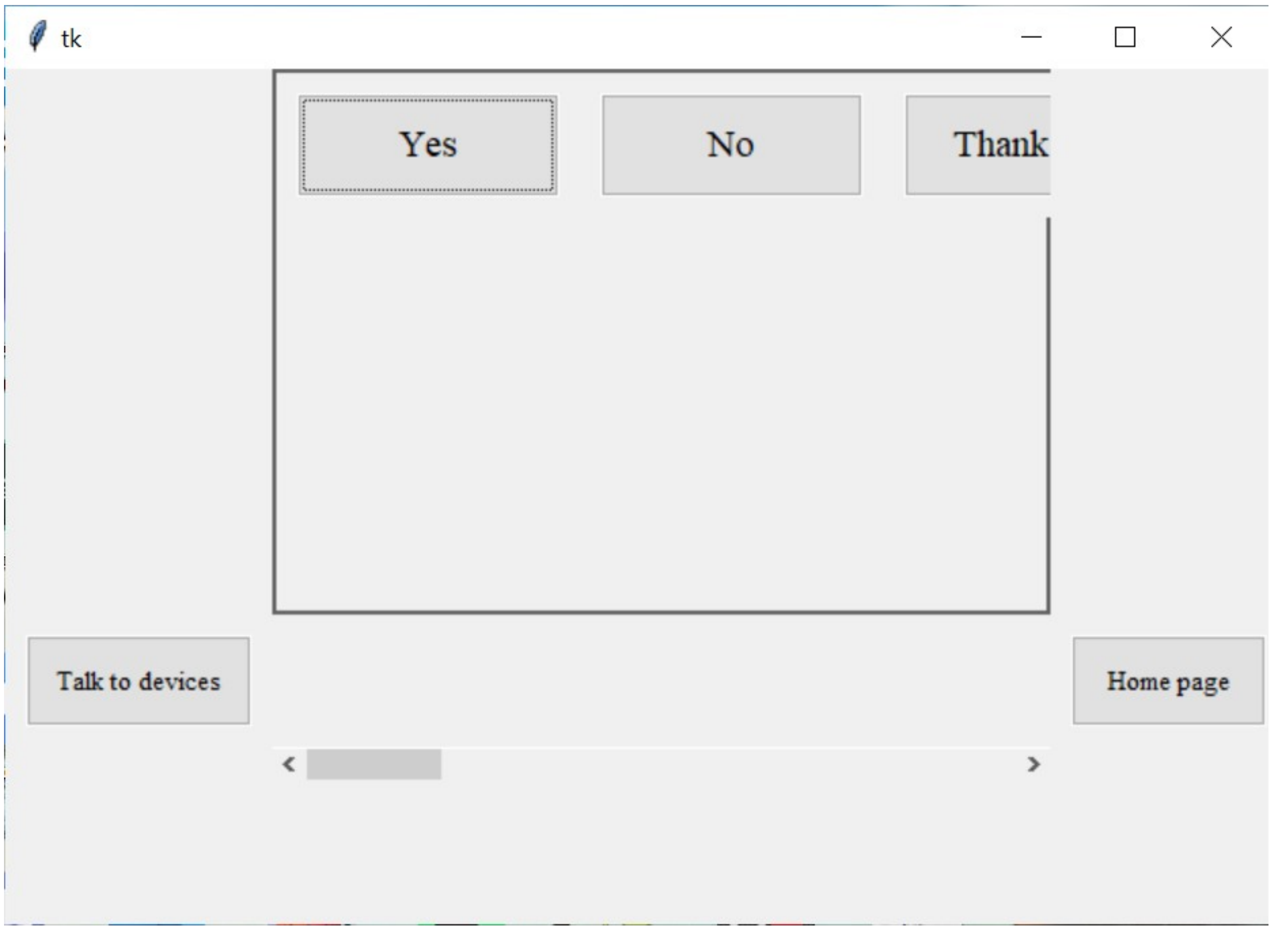
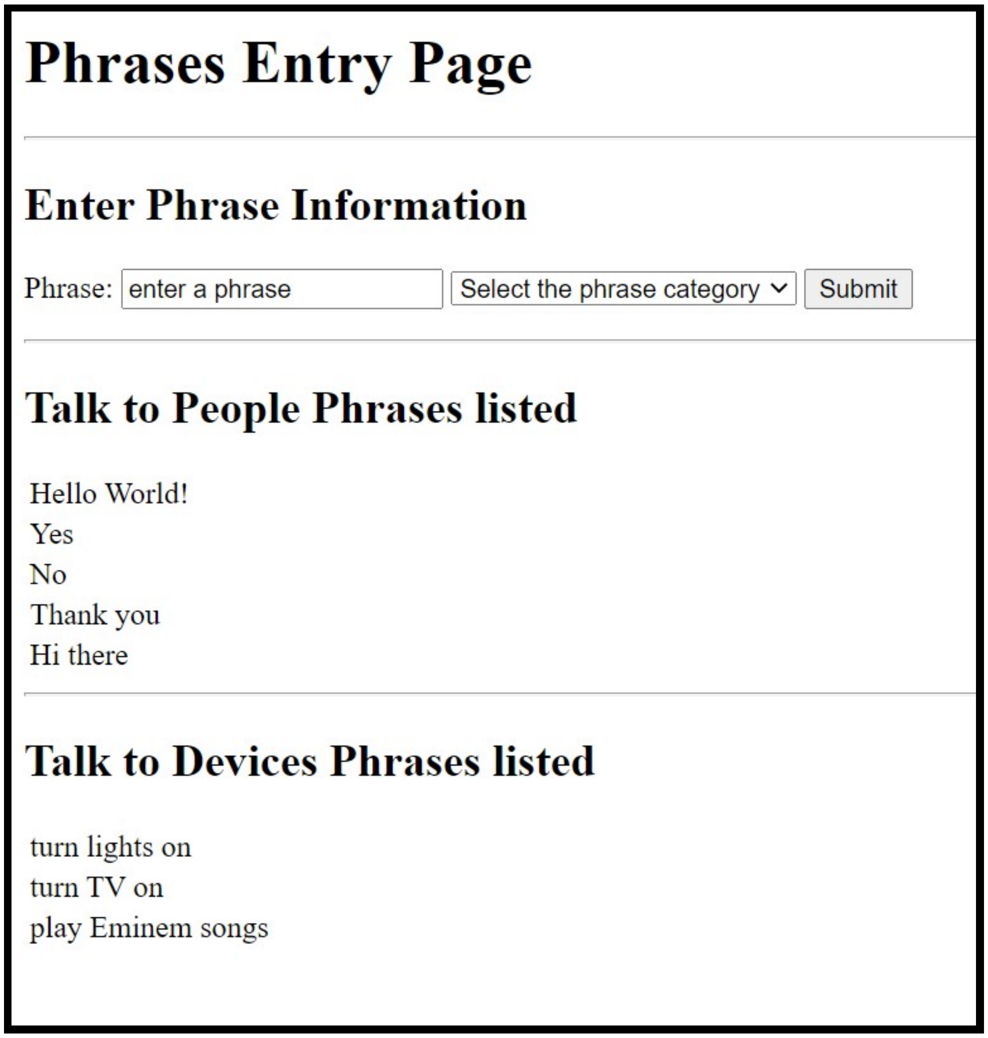


Figure 4-6: Highlighted button in the talk to people category

When the user wishes to go back to home screen to a different category, they can move the joystick downwards and to the left or right to select either Talk to devices or homepage.

4.3 Adding Phrases

If the user would like to add a new phrase their caregiver can access the following page: http://zainabbadawy.com/index.php, the following page can be accessed from any device, anytime, and anywhere.



Phrases Entry Page

Enter Phrase Information

Phrase: enter a phrase | Select the phrase category | Submit

Talk to People Phrases listed

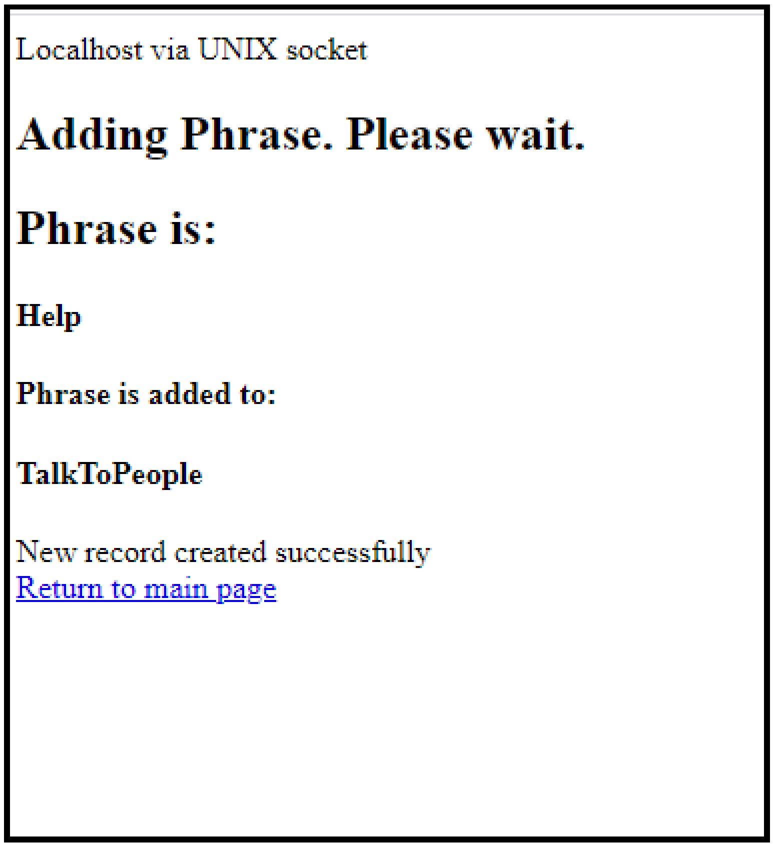
Hello World!
Yes
No
Thank you
Hi there

Talk to Devices Phrases listed

turn lights on
turn TV on
play Eminem songs

Figure 4-7:Main caregiver interface

After the user adds a new phrase, they will be redirected to the page in figure 4.8: Confirmation of adding a new phrase.



Localhost via UNIX socket

Adding Phrase. Please wait.

Phrase is:

Help

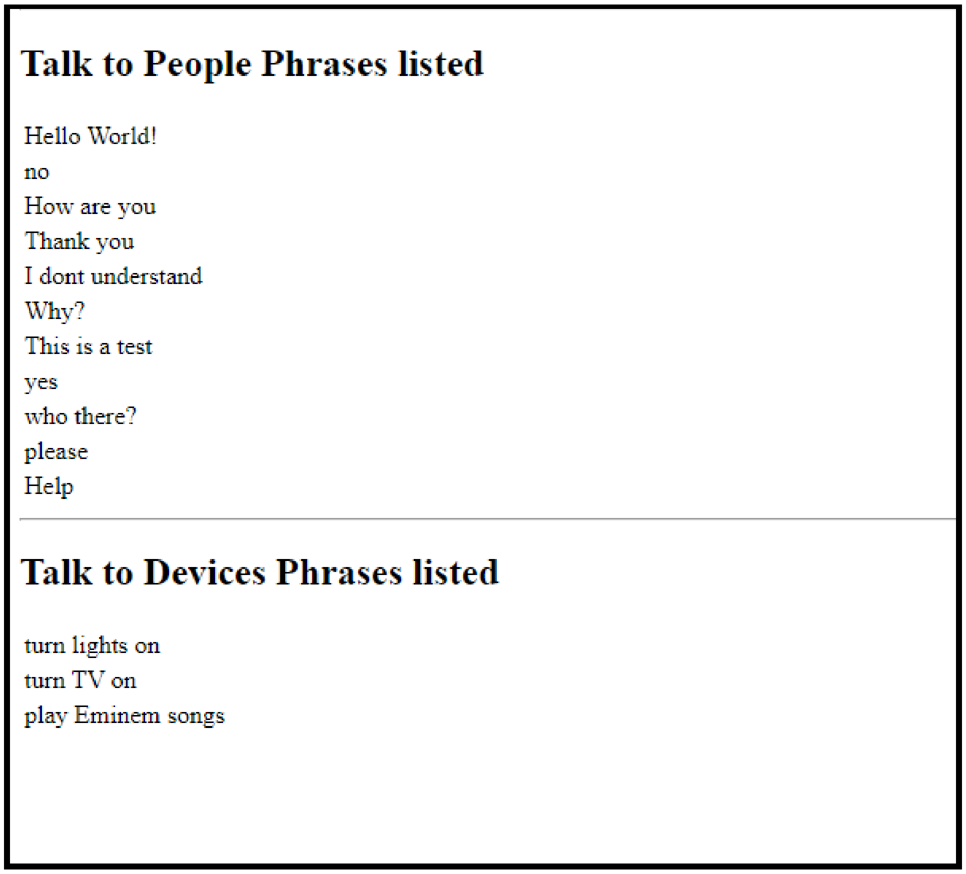
Phrase is added to:

TalkToPeople

New record created successfully
Return to main page

Figure 4-8: Confirmation of adding a new phrase

Finally, the main page will be updated with the recently added phrase.



Talk to People Phrases listed

Hello World!
no
How are you
Thank you
I dont understand
Why?
This is a test
yes
who there?
please
Help

Talk to Devices Phrases listed

turn lights on
turn TV on
play Eminem songs

Figure 4-9: Main page will be updated with the recently added phrase

5 Troubleshooting & Support

5.1 Error Messages or Behaviors

Some examples of errors that may occur, but are not limited to, are the wordlist may not update properly when database is updated, python app doesn’t run properly/times out, buttons in the python app aren’t selected properly, and the buttons could be highlighted with a colour that is difficult to read. There are multiple suggestions regarding how to fix these issues.

5.2 Special Considerations

First off, check if wifi connection is stable enough for the Pi to access the database in order for the python app to be able to update. Furthermore, if the program showcases any hiccups, attempt to restart the program and if the issue persists, try restarting the system. After all the troubleshooting, if there is still a problem, a technician may be needed. The problem at this point may relate to anything from driver or software updates, which can change software compatibility, to a hardware malfunction, which may need part replacement.

5.3 Maintenance

Check for damaged power cables regularly. If there is damage, please replace the damaged cable(s) immediately. The casing for the Pi and the display should be cleaned every few months (5-6 months recommended). Any extra data stored onto the Raspberry Pi’s SD card should be backed up externally, in case of a system failure and/or file corruption.

5.4 Support

Both the web service for updating the word/phrase list and the SuperSpeak app will contain contact information at the bottom of the page, it will include an email address with a custom domain. There will also be a separate interface for a caretaker to submit a ticket with regards to what the issue is.

6 Product Documentation

SuperSpeak is a project with a mix of hardware and software included. The main hardware components include a Raspberry Pi, analogue-to-digital converter, speaker, display, and a joystick; whilst the software includes a python application and a web-based interface to update the python application's database. For the final prototype, the joystick is connected to the Pi via analogue-to-digital converter, and is used as the main input interface for the python app. On the Pi, Raspbian is used for its OS and the python application is run on it. The python application is a GUI where it shows an array of buttons. On the first panel on the app, the speech category is picked which then the user is prompted with another array of buttons that they can select from. These buttons are meant to show the text and phrases that are dedicated to the needs and requests of an individual user. The text also has an image accompanied with it to illustrate the meaning of that word or phrase. The overall feasibility of this prototype is high due to the various parts used. The two core components, the Pi and Python app are almost essential to the final design. The reasoning behind is that the Pi provides just the right amount of processing power and offers portability, I/O ports, and graphical interface (via an OS); the python app is also essential because python as a programming language is currently widely used and known, offers the most flexibility for connecting api's, implementation in web environments, and less steep learning curve, which all this makes it easier for whomever to work and upgrade/update the app. The joystick, speaker, and display are essential to have, but their sizes and shapes can vary since each user may have different needs of measurement for the said tools. This prototype is for the purpose of being able to showcase the core functionality of the project which is a user picking a word from a database and saying that word out loud, to be put simply.

6.1 Final Prototype

6.1.1 BOM (Bill of Materials)

Component	Price	Link

Raspberry Pi 3 - Model A+ (PLUS) **	$32.95 + tax + shipping	https://www.buyapi.ca/product/raspberry-pi-3-model-a-plus-512mb-ram/
USB Powered Speakers **	$13.95 +tax + shipping	https://www.buyapi.ca/product/usb-powered-speakers/
Arcade Joystick	$ 32.99	https://www.amazon.ca/gp/product/B01M2X88QP/ref=ppx_yo_dt_b_asin_title_o02_s00?ie=UTF8&psc=1
Raspberry Pi Touchscreen Monitor	$ 23.99	https://www.amazon.ca/gp/product/B07SKVF392/ref=ppx_yo_dt_b_asin_title_o07_s00?ie=UTF8&psc=1
Samsung EVO Plus 32GB sd card	$11.99 +tax	https://www.bestbuy.ca/en-ca/product/samsung-evo-plus-32gb-95mb-s-microsdhc-uhs-1-memory-card/10717558
I2C LCD screen *	$0	Already had one
HDMI cable	$0	Already had one
keyboard	$0	Already had one
Mouse	$0	Already had one
Monitor screen	$0	Already had one
Joystick	$0	Already had a pack

Arcade low force Push button *	$0	Already had a pack
Jumper cables	$0	Already had a pack
PLA and 3D printing	$0	Already have a 3D printer
PCB boards	$0	Already have a pack
Elastic straps	$0	Borrowed some from my mom

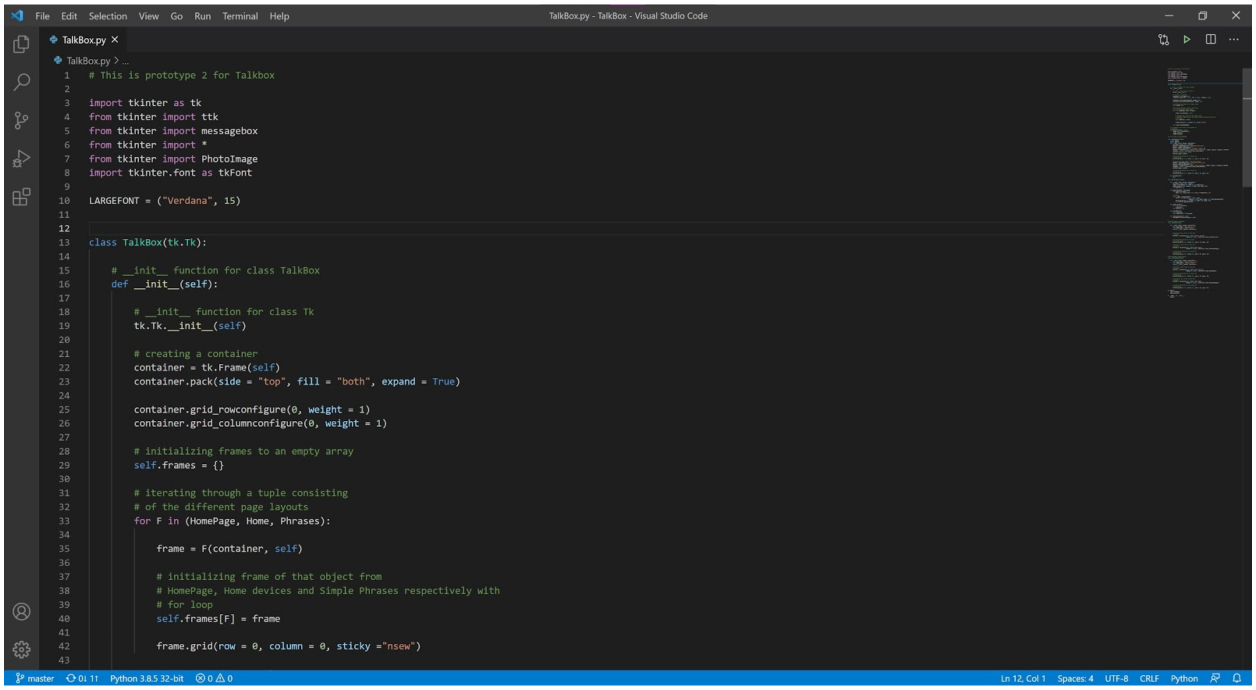
6.1.2 Equipment list

There was a moderate usage of 3D printers for printing out a chassis for the joystick, display, and the Pi. Soldering tool was also used to connect the joystick to the digital-to-analogue converter and then to the Raspberry Pi microcontroller. A PC was also needed to be able to code the python program that is supposed to run on the Pi. An online database was also used in order to allow updates for future words and phrases to be added to the word list of the python app.

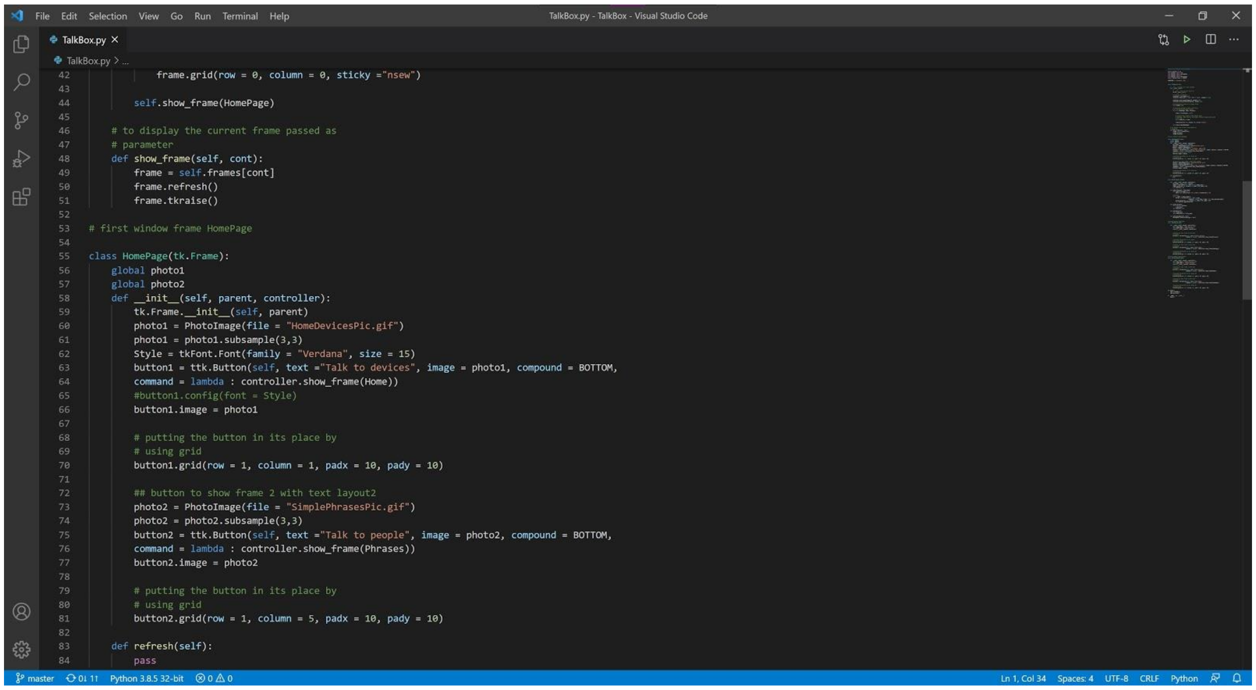
6.2 Instructions

6.2.1 Software:

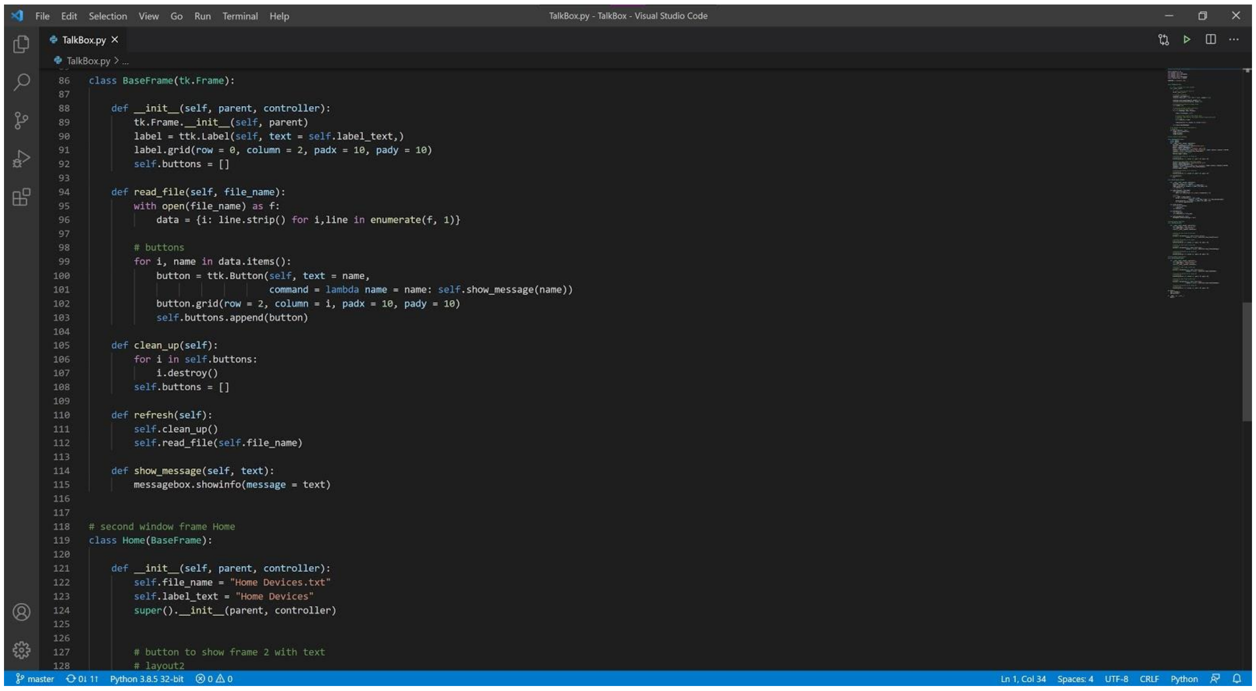
In terms of the software prototype, we have created a python script that creates a graphical user interface with the Tkinter package. We have created 5 different classes that interact with each other: the TalkBox class, the HomePage class, the BaseFrame class, the Home class and the Phrases class. The TalkBox class acts as the initializer for the frame of the entire program which creates the frames for every page created in the class. The HomePage class contains two widget buttons that have photo images, where their commands are to go to their respective pages: “Talk to devices” which opens the Home class and “Talk to people” which opens the Phrases class. The most important class is the BaseFrame class which acts as a parent to both the Home class and the Phrases class since it contains all the common functions between both classes. This class contains the function that reads the text files and creates widget buttons from all the entries from those files. The commands for those buttons are currently creating a message show box that returns the specific sentence selected. For the future, we are planning to replace those current commands with code to send to the text-to-speech API which will voice those sentences. In our BaseFrame, we also have a refresh function which allows the pages to get automatically updated with buttons when the text files are updated, without having to kill the program.


File Edit Selection View Go Run Terminal Help
TalkBox.py - TalkBox - Visual Studio Code
TalkBox.py
TalkBox.py > ...
master
Python 3.8.5 32-bit
Ln 12, Col 1   Spaces: 4   UTF-8   CRLF   Python

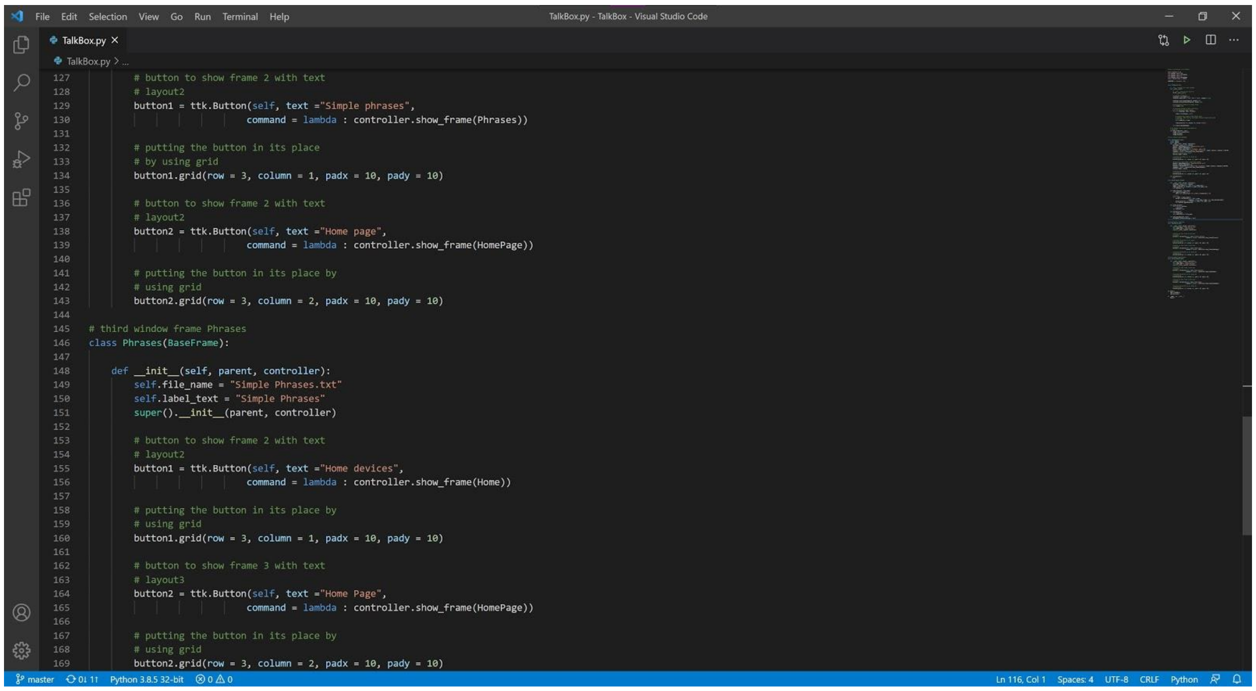
```python
# This is prototype 2 for Talkbox

import tkinter as tk
from tkinter import ttk
from tkinter import messagebox
from tkinter import *
from tkinter import PhotoImage
import tkinter.font as tkFont

LARGEFONT = ("Verdana", 15)


class TalkBox(tk.Tk):

    # __init__ function for class TalkBox
    def __init__(self):

        # __init__ function for class Tk
        tk.Tk.__init__(self)

        # creating a container
        container = tk.Frame(self)
        container.pack(side = "top", fill = "both", expand = True)

        container.grid_rowconfigure(0, weight = 1)
        container.grid_columnconfigure(0, weight = 1)

        # initializing frames to an empty array
        self.frames = {}

        # iterating through a tuple consisting
        # of the different page layouts
        for F in (HomePage, Home, Phrases):

            frame = F(container, self)

            # initializing frame of that object from
            # HomePage, Home devices and Simple Phrases respectively with
            # for loop
            self.frames[F] = frame

            frame.grid(row = 0, column = 0, sticky ="nsew")

```



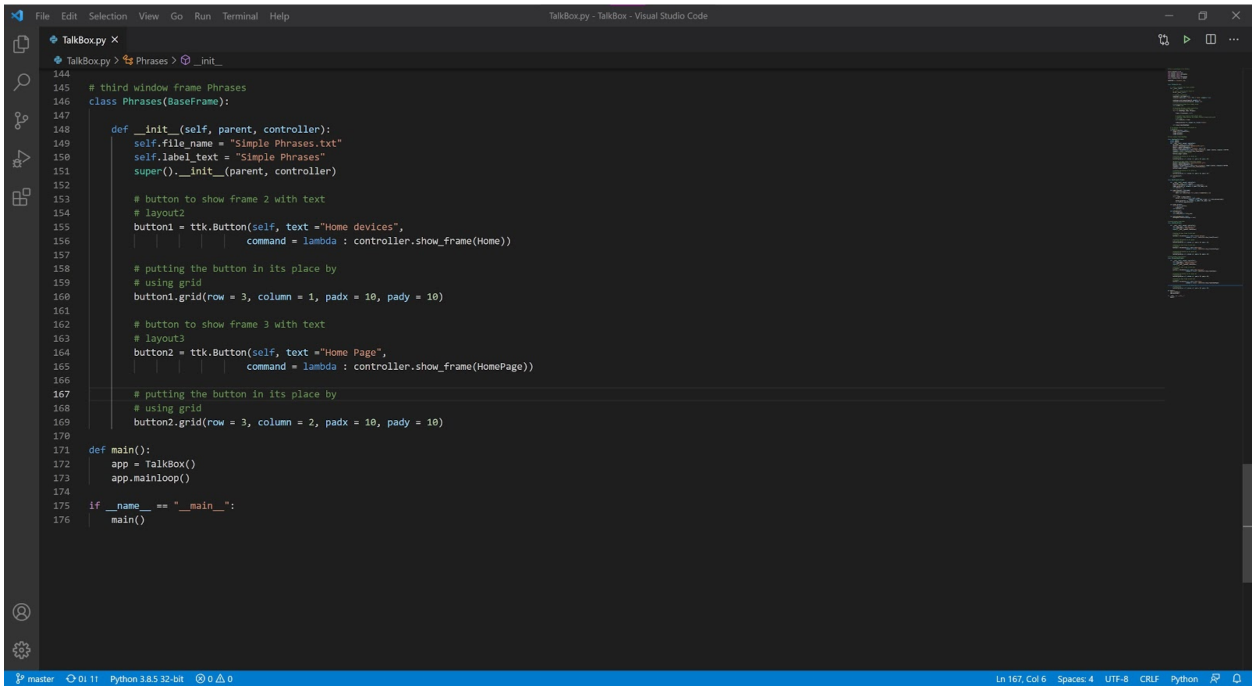
```python
            frame.grid(row = 0, column = 0, sticky ="nsew")

        self.show_frame(HomePage)

    # to display the current frame passed as
    # parameter
    def show_frame(self, cont):
        frame = self.frames[cont]
        frame.refresh()
        frame.tkraise()

# first window frame HomePage

class HomePage(tk.Frame):
    global photo1
    global photo2
    def __init__(self, parent, controller):
        tk.Frame.__init__(self, parent)
        photo1 = PhotoImage(file = "HomeDevicesPic.gif")
        photo1 = photo1.subsample(3,3)
        Style = tkFont.Font(family = "Verdana", size = 15)
        button1 = ttk.Button(self, text ="Talk to devices", image = photo1, compound = BOTTOM,
        command = lambda : controller.show_frame(Home))
        #button1.config(font = Style)
        button1.image = photo1

        # putting the button in its place by
        # using grid
        button1.grid(row = 1, column = 1, padx = 10, pady = 10)

        ## button to show frame 2 with text layout2
        photo2 = PhotoImage(file = "SimplePhrasesPic.gif")
        photo2 = photo2.subsample(3,3)
        button2 = ttk.Button(self, text ="Talk to people", image = photo2, compound = BOTTOM,
        command = lambda : controller.show_frame(Phrases))
        button2.image = photo2

        # putting the button in its place by
        # using grid
        button2.grid(row = 1, column = 5, padx = 10, pady = 10)

    def refresh(self):
        pass
```



```python
class BaseFrame(tk.Frame):

    def __init__(self, parent, controller):
        tk.Frame.__init__(self, parent)
        label = ttk.Label(self, text = self.label_text,)
        label.grid(row = 0, column = 2, padx = 10, pady = 10)
        self.buttons = []

    def read_file(self, file_name):
        with open(file_name) as f:
            data = {i: line.strip() for i,line in enumerate(f, 1)}

        # buttons
        for i, name in data.items():
            button = ttk.Button(self, text = name,
                                command = lambda name = name: self.show_message(name))
            button.grid(row = 2, column = i, padx = 10, pady = 10)
            self.buttons.append(button)

    def clean_up(self):
        for i in self.buttons:
            i.destroy()
        self.buttons = []

    def refresh(self):
        self.clean_up()
        self.read_file(self.file_name)

    def show_message(self, text):
        messagebox.showinfo(message = text)


# second window frame Home
class Home(BaseFrame):

    def __init__(self, parent, controller):
        self.file_name = "Home Devices.txt"
        self.label_text = "Home Devices"
        super().__init__(parent, controller)


        # button to show frame 2 with text
        # layout2
```


File Edit Selection View Go Run Terminal Help
TalkBox.py - TalkBox - Visual Studio Code
TalkBox.py
TalkBox.py > ...
master
Python 3.8.5 32-bit
Ln 116, Col 1 Spaces: 4 UTF-8 CRLF Python

```python
        # button to show frame 2 with text
        # layout2
        button1 = ttk.Button(self, text ="Simple phrases",
                                command = lambda : controller.show_frame(Phrases))

        # putting the button in its place
        # by using grid
        button1.grid(row = 3, column = 1, padx = 10, pady = 10)

        # button to show frame 2 with text
        # layout2
        button2 = ttk.Button(self, text ="Home page",
                                command = lambda : controller.show_frame(HomePage))

        # putting the button in its place by
        # using grid
        button2.grid(row = 3, column = 2, padx = 10, pady = 10)

# third window frame Phrases
class Phrases(BaseFrame):

    def __init__(self, parent, controller):
        self.file_name = "Simple Phrases.txt"
        self.label_text = "Simple Phrases"
        super().__init__(parent, controller)

        # button to show frame 2 with text
        # layout2
        button1 = ttk.Button(self, text ="Home devices",
                                command = lambda : controller.show_frame(Home))

        # putting the button in its place by
        # using grid
        button1.grid(row = 3, column = 1, padx = 10, pady = 10)

        # button to show frame 3 with text
        # layout3
        button2 = ttk.Button(self, text ="Home Page",
                                command = lambda : controller.show_frame(HomePage))

        # putting the button in its place by
        # using grid
        button2.grid(row = 3, column = 2, padx = 10, pady = 10)
```


File Edit Selection View Go Run Terminal Help
TalkBox.py - TalkBox - Visual Studio Code
TalkBox.py
TalkBox.py > Phrases > __init__
master
Python 3.8.5 32-bit
Ln 167, Col 6 Spaces: 4 UTF-8 CRLF Python

```python

# third window frame Phrases
class Phrases(BaseFrame):

    def __init__(self, parent, controller):
        self.file_name = "Simple Phrases.txt"
        self.label_text = "Simple Phrases"
        super().__init__(parent, controller)

        # button to show frame 2 with text
        # layout2
        button1 = ttk.Button(self, text ="Home devices",
                                command = lambda : controller.show_frame(Home))

        # putting the button in its place by
        # using grid
        button1.grid(row = 3, column = 1, padx = 10, pady = 10)

        # button to show frame 3 with text
        # layout3
        button2 = ttk.Button(self, text ="Home Page",
                                command = lambda : controller.show_frame(HomePage))

        # putting the button in its place by
        # using grid
        button2.grid(row = 3, column = 2, padx = 10, pady = 10)

def main():
    app = TalkBox()
    app.mainloop()

if __name__ == "__main__":
    main()
```

Figure 6-0-1: Python script that creates the GUI using Tkinter

6.2.2 Hardware:

1. Initialization

For the initialization process, the Raspberry Pi Imager was used to flash the SD card with the "Raspberry Pi OS (32-bit)". Following that, the Pi was connected to a monitor using a HDMI cable and the country, language, time zone, password, and wifi were set up. More details can be found on Raspberry pi official website under Setting up your Raspberry Pi.

The TalkBox github repository was cloned to the Raspberry Pi, and the talkbox.py python script was successfully executed. The talkbox.py script outputs the following panel, where the user can first choose if they want to output phrases that talk to people or talk to devices as shown in the figure "User main display panel". After the user selects one of the options, they will be directed to another panel that provides multiple buttons with phases in the database, as shown in the figure '"Home Devices" phrases displayed as buttons, ready to be voiced via speaker', and figure '"Simple Phrases" word bank displayed as buttons', ready to be voiced via speaker.

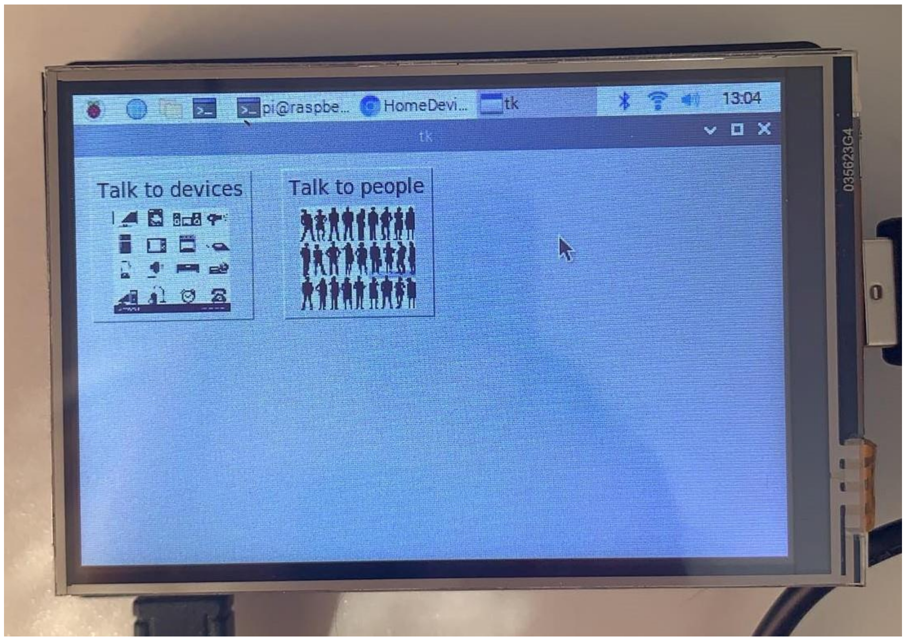


Figure 6-0-2: User main display panel

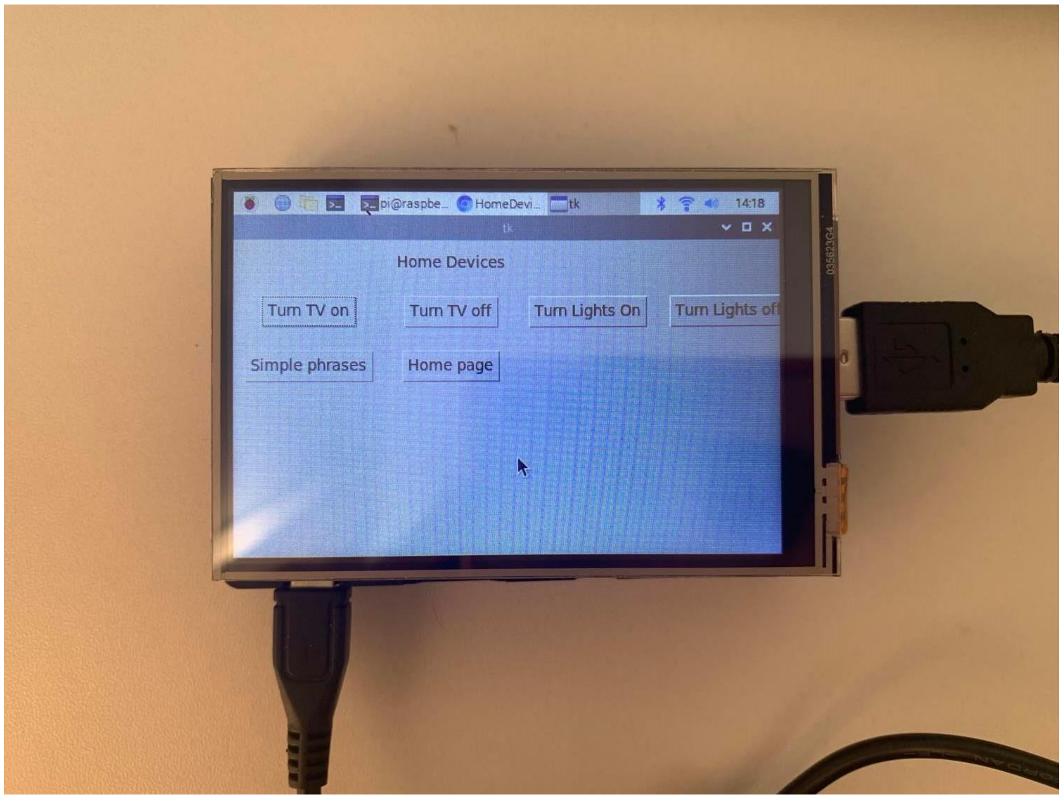


Figure 6-0-3: “Home Devices” phrases displayed as buttons, ready to be voiced via speaker

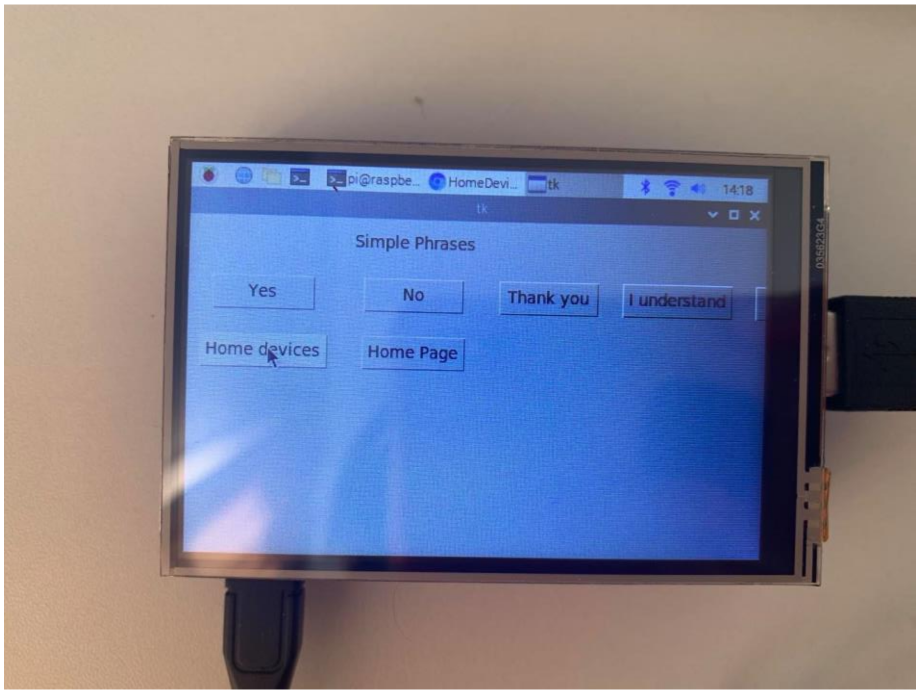


Figure 6-0-4:“Simple Phrases” word bank displayed as buttons, ready to be voiced via speaker

2. MYSQL Database

The team used MYSQL Database Wizard on Godaddy cpanel to set up the MYSQL Database, as shown under Databases in cpanel main page in Figures 6.1.3.1. The team then typed the new database name and users credentials as shown in the

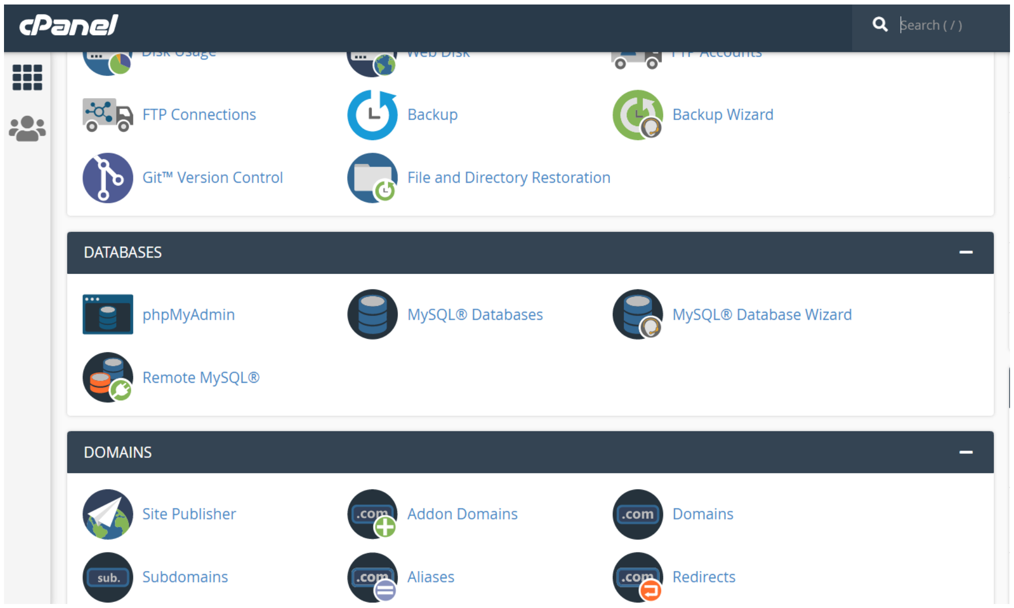
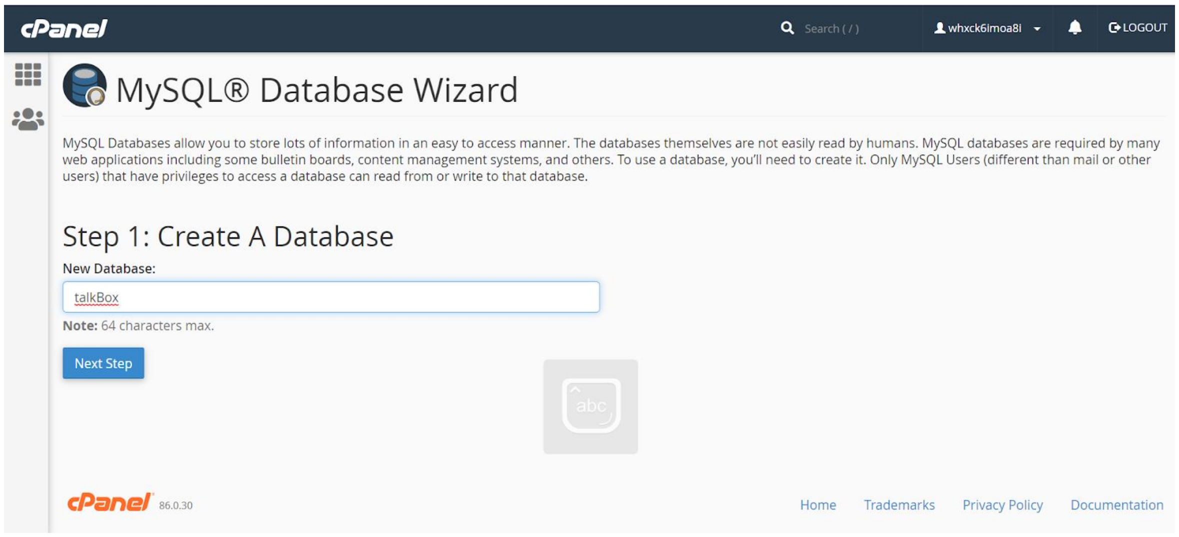


Figure 6-0-5:cPanel main page



MySQL® Database Wizard

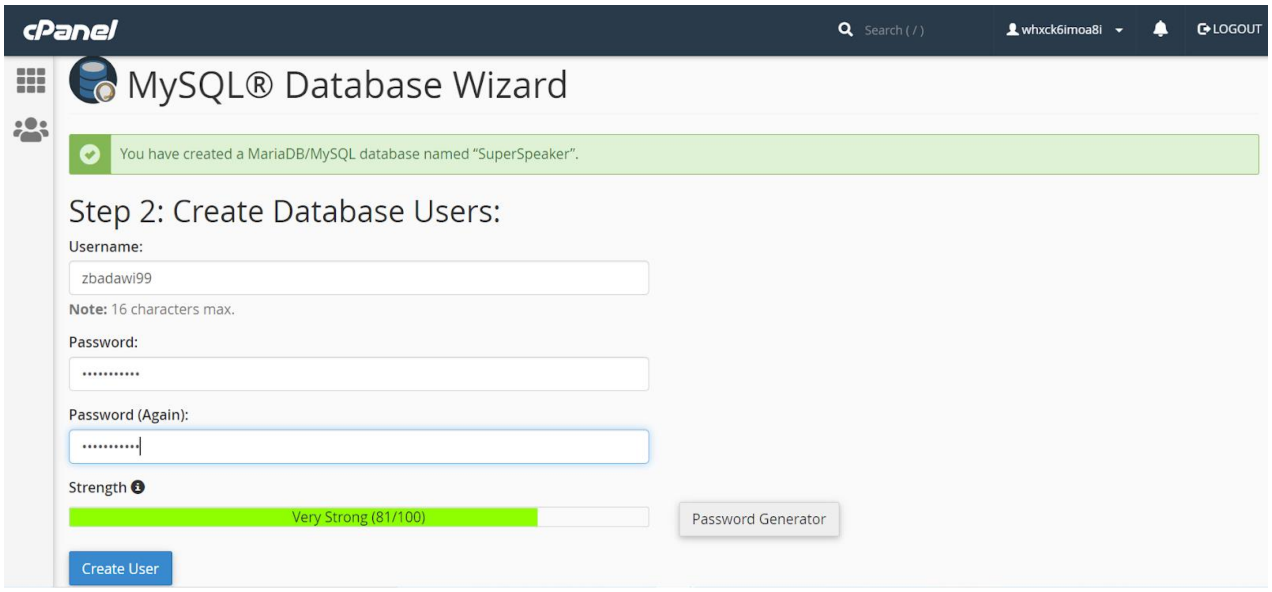
MySQL Databases allow you to store lots of information in an easy to access manner. The databases themselves are not easily read by humans. MySQL databases are required by many web applications including some bulletin boards, content management systems, and others. To use a database, you'll need to create it. Only MySQL Users (different than mail or other users) that have privileges to access a database can read from or write to that database.

Step 1: Create A Database

New Database: talkBox

Note: 64 characters max.

Figure 6-0-6: Step1: Create A Database



MySQL® Database Wizard

You have created a MariaDB/MySQL database named “SuperSpeaker”.

Step 2: Create Database Users:

Username: zbadawi99

Note: 16 characters max.

Password: •••••••••••

Password (Again): •••••••••••

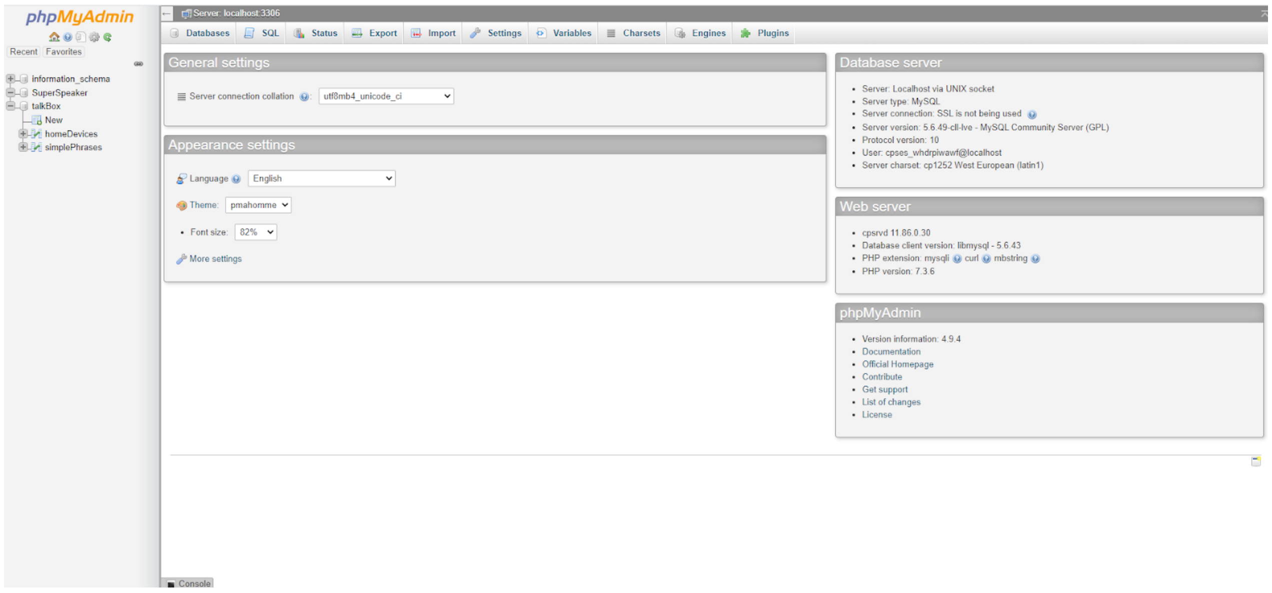
Strength

Very Strong (81/100)

Figure 6-0-7: Step2: Create A Database Users

3. PHPMyAdmin

From the cPanel main page the phpMyAdmain gooey can be accessed where you can edit the sql database created in the previous step.



General settings

Server connection collation: utf8mb4_unicode_ci

Appearance settings

Language: English

Theme: pmahomme

- Font size: 82%

More settings

Database server

- Server: Localhost via UNIX socket
- Server type: MySQL
- Server connection: SSL is not being used
- Server version: 5.6.49-cll-lve - MySQL Community Server (GPL)
- Protocol version: 10
- User: cpses_whdrpiwawf@localhost
- Server charset: cp1252 West European (latin1)

Web server

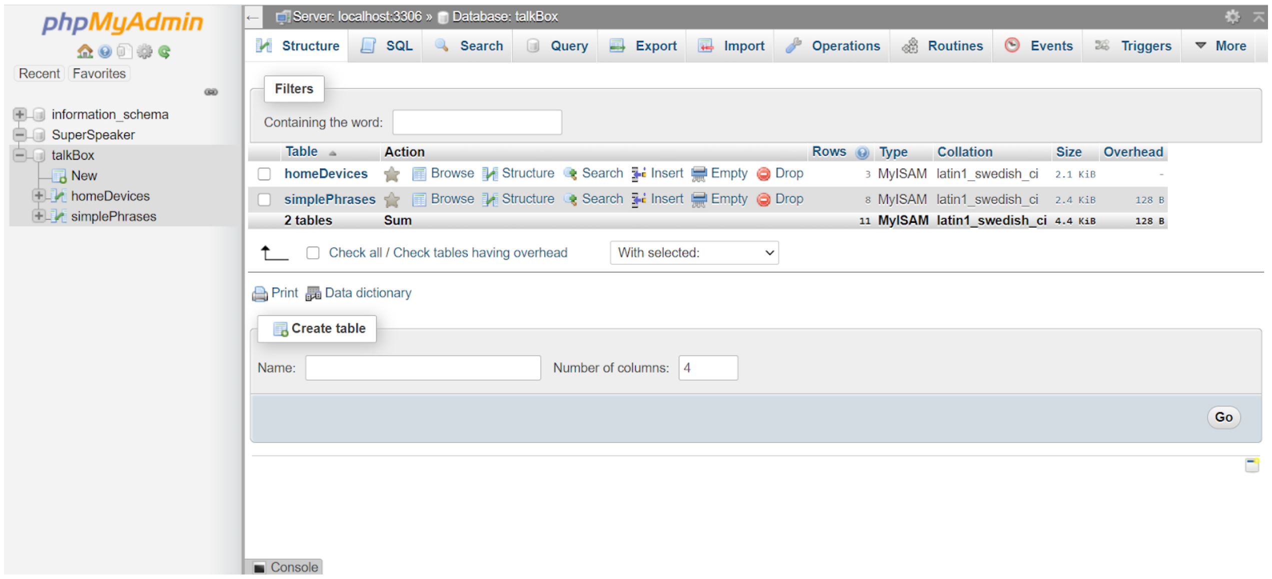
- cpsrvd 11.86.0.30
- Database client version: libmysql - 5.6.43
- PHP extension: mysqli curl mbstring
- PHP version: 7.3.6

phpMyAdmin

- Version information: 4.9.4
- Documentation
- Official Homepage
- Contribute
- Get support
- List of changes
- License

Figure 6-0-8:PhpMyAdmin to main page

Two tables were created, “homeDevices” and “simplePhrases” and were populated with phrases that are directed to either home assistant devices or people.



Containing the word:

	Table	Action	Rows	Type	Collation	Size	Overhead
☐	homeDevices	Browse Structure Search Insert Empty Drop	3	MyISAM	latin1_swedish_ci	2.1 KiB	-
☐	simplePhrases	Browse Structure Search Insert Empty Drop	8	MyISAM	latin1_swedish_ci	2.4 KiB	128 B
	2 tables	Sum	11	MyISAM	latin1_swedish_ci	4.4 KiB	128 B

Check all / Check tables having overhead   With selected:

Print Data dictionary

Create table

Name: Number of columns: 4

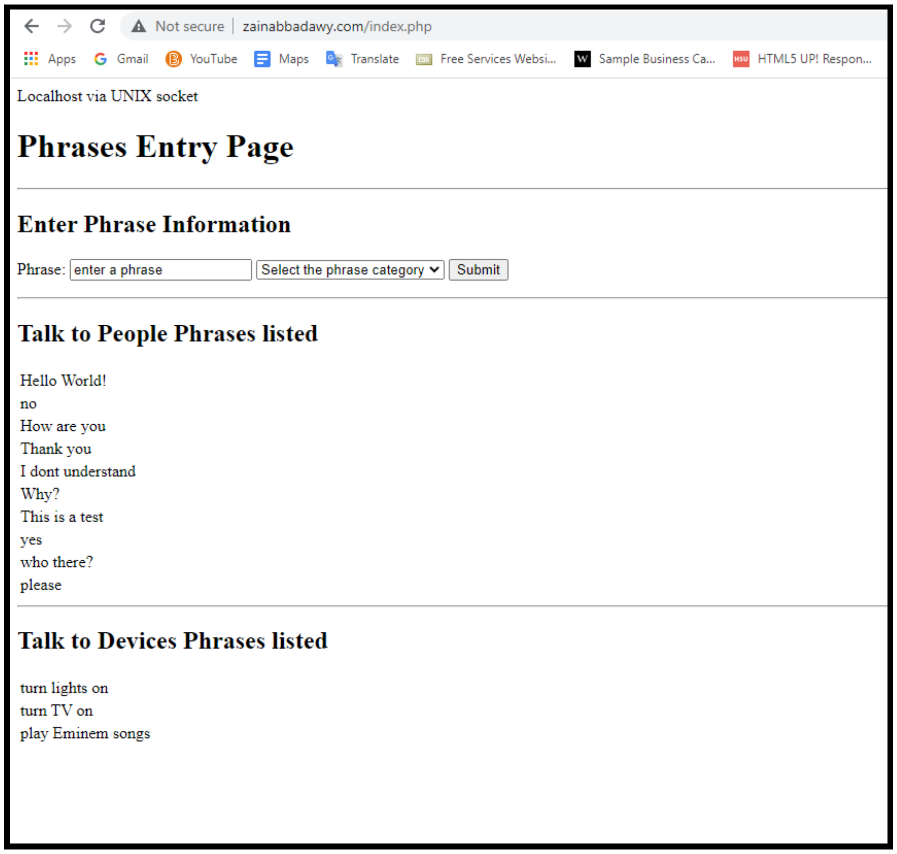
Go

Figure 6-0-9:Editing the database

4. Create an HTML input form for a PHP web app

The team coded 4 php files that will do the following three functions:

- In the index.php file: is the main file that connects db_connect.php, db_insert.php, db_list_all_results.php. Its main purpose is to allow the user (caregiver) to input phrases to the talkBox database in a form of a box and ask them to choose the category of the phrases: talk to people or talk to devices. The output of this code is shown in the following figure:
- In the db_connect.php file: it connects the mySQL database to php.
- in db_insert.php: the function of this file is to insert the new phrase added to the correct table either the simplePhrases or the homeDevices and it outputs a message to the user if the phase has been added successfully. The following code outputs the following page



Localhost via UNIX socket

Phrases Entry Page

Enter Phrase Information

Phrase: enter a phrase | Select the phrase category | Submit

Talk to People Phrases listed

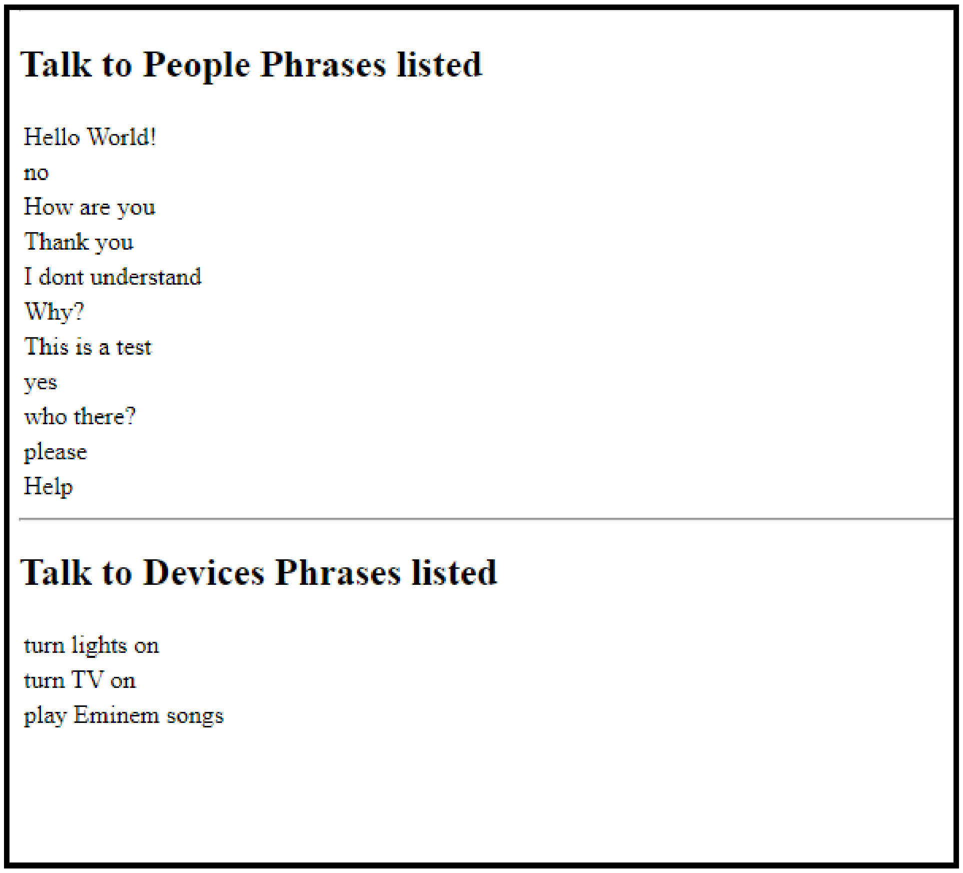
Hello World!
no
How are you
Thank you
I dont understand
Why?
This is a test
yes
who there?
please

Talk to Devices Phrases listed

turn lights on
turn TV on
play Eminem songs

Figure 6-0-10: Main Page

- db_list_all_results.php: retrieves all the all the phrases inputted in the database and lists it in a table form. The output of the following file is shown in figure



Talk to People Phrases listed

Hello World!
no
How are you
Thank you
I dont understand
Why?
This is a test
yes
who there?
please
Help

Talk to Devices Phrases listed

turn lights on
turn TV on
play Eminem songs

Figure 6-0-11: Editing page

The code for the following files can be found in Appendix II.

Finally, the USB joystick was connected to the raspberry pi to eliminate the use of the keyboard keys and the AntiMicro app was used to set up its movements as shown in section 4.1.

6.3 Testing & Validation

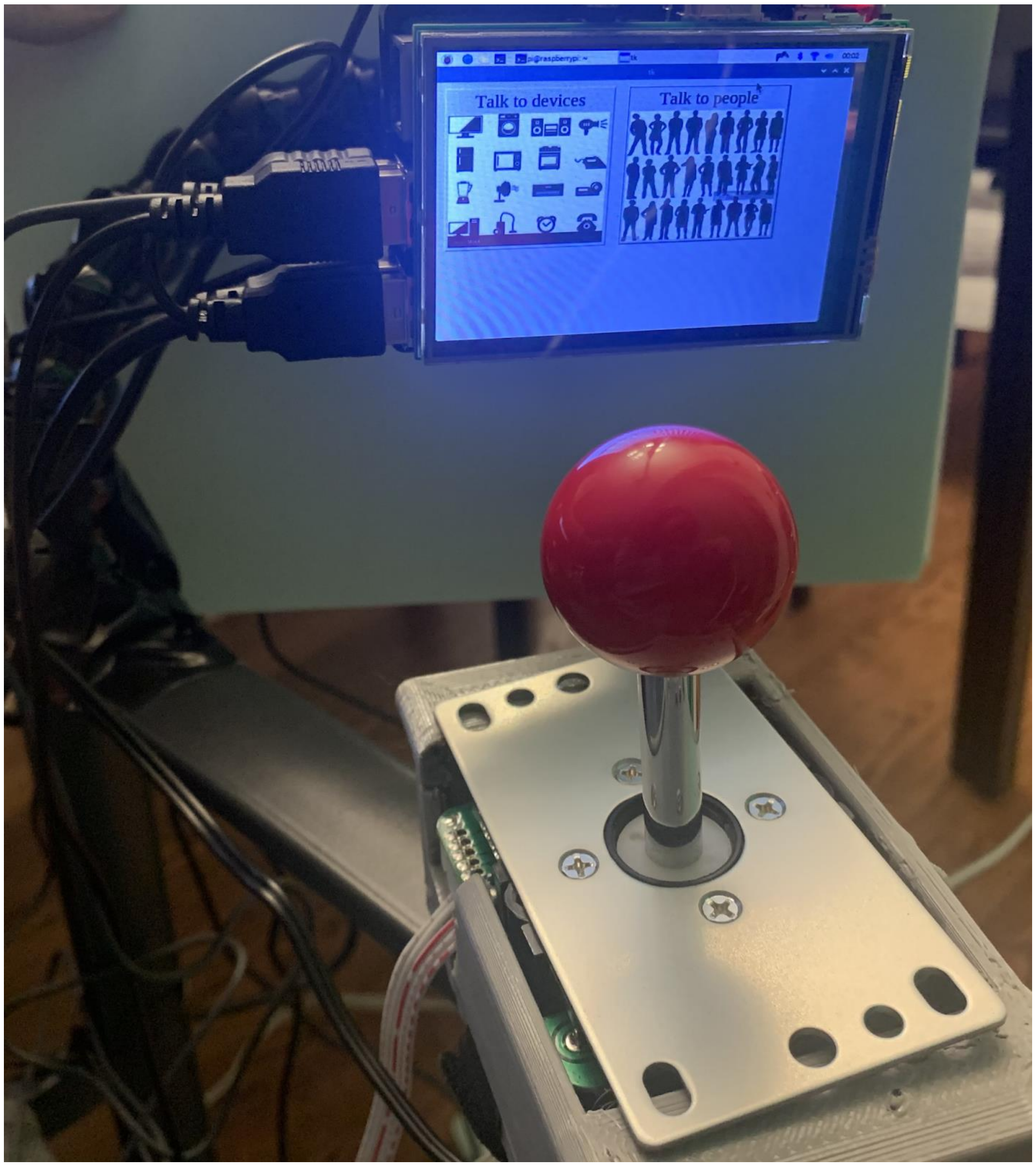


Figure 6-12: Hardware and Software prototype parts

The test result of the hardware prototype’s joystick querying data from MySQL, and then sending commands to the speaker to output phrases were successful. We were able to test the joystick response time and sound quality produced by the speakers were adequate in our prototype testing. Sounds were tested to communicate with a siri assistant and ensure that the SuperSpeak can produce well enough sound to awaken smart home assistants. Currently, the joystick and the screen are designed to be mounted on either arm of the wheelchair, and a gooseneck mount is used for the screen to allow the user to comfortably view the screen from any angle. The weight of the device is the sum of the Raspberry Pi and the screen is about 200 grams which is light enough to be mounted on any wheelchararm. Additionally the joystick mount is designed to fit a velcro strap that will wrap tightly around the wheelchair arms.

In terms of the software prototype, we were able to measure the user friendliness of our interface. We believe in terms of appearance and ease of use for the user, that our current prototype is still not up to par since the font is quite small, therefore it is hard to read. The frame of the GUI also changes sizes from switching between reading small and large text files since the buttons are created in a row, which is quite displeasing. The quality of images displayed on the screen of the user was also evaluated and they were clear and visually pleasing and added a nice effect to our home page. Although we couldn’t obtain a hard value for the image quality, we were able to judge them visually. Unfortunately, due to the small screen size, we were not able to see all the buttons that were created, which is an issue that will be addressed in the future prototype by allowing only 2-3 buttons per page and adding the functionality of going between different pages which would encompass the different options of buttons available to the user.

7 Conclusions and Recommendations for Future Work

This project ended up being quite successful in the aspect of what we had envisioned from the start. We were able to demonstrate a device that offers versatility in terms of future expansion while still offering a functional device that helps its user say what they want to, with the ability to expand their word list as well. Throughout the project, a lot of barriers were met such as hardware compatibility and software bugs, but in the end, the final prototype managed achieving almost everything the final should offer based on original design concepts. For future work, the goal is to make a more user friendly web page for the caretaker, while for the main user, the SuperSpeak app would offer color blind mode, different font and text sizes, and the ability to support multiple languages.

8 Bibliography

https://www.raspberrypi.org/documentation/configuration/warning-icons.md

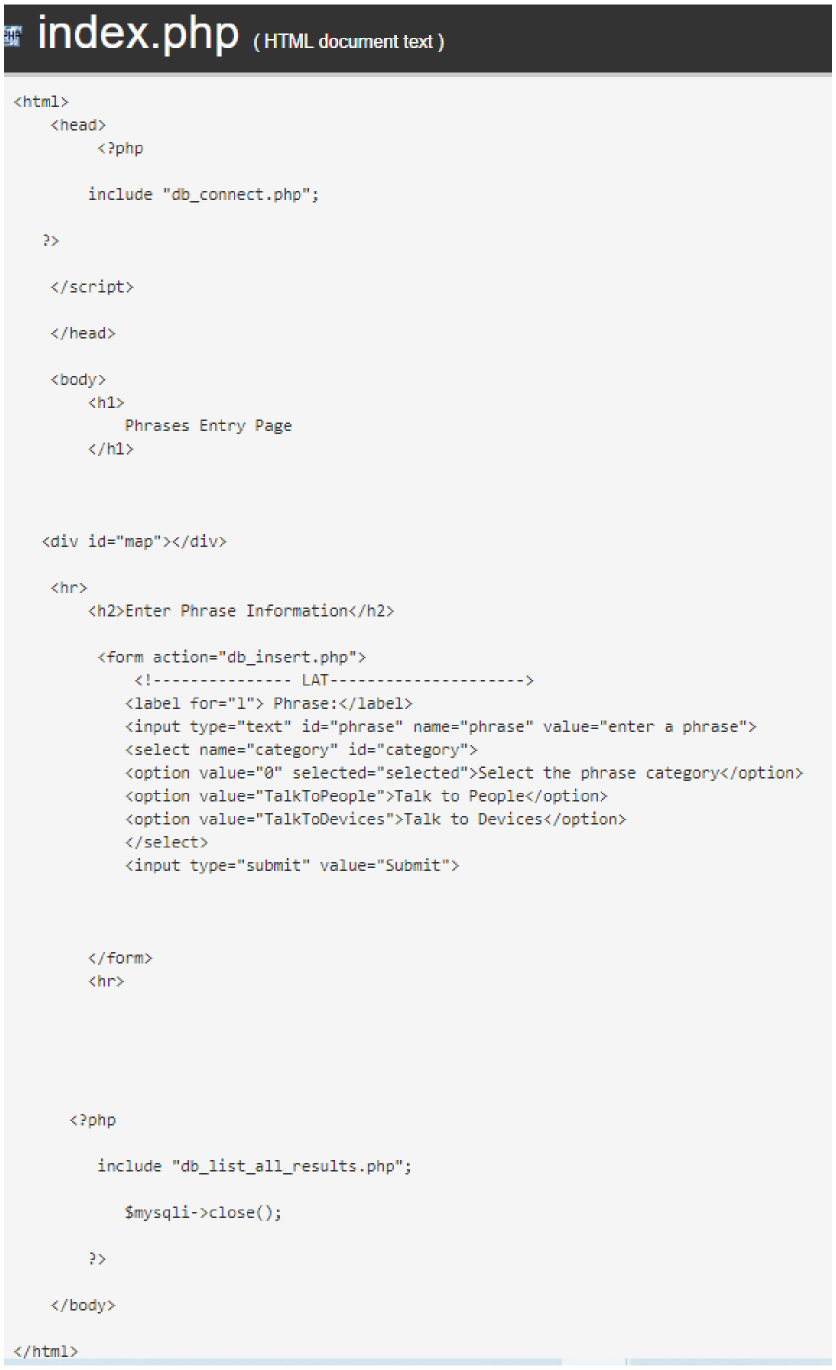
Projects.raspberrypi.org, projects.raspberrypi.org/en/projects/raspberry-pi-setting-up.

Emmet. “Setup a Raspberry Pi MYSQL Database.” Pi My Life Up, 9 Feb. 2021, pimylifeup.com/raspberry-pi-mysql/.

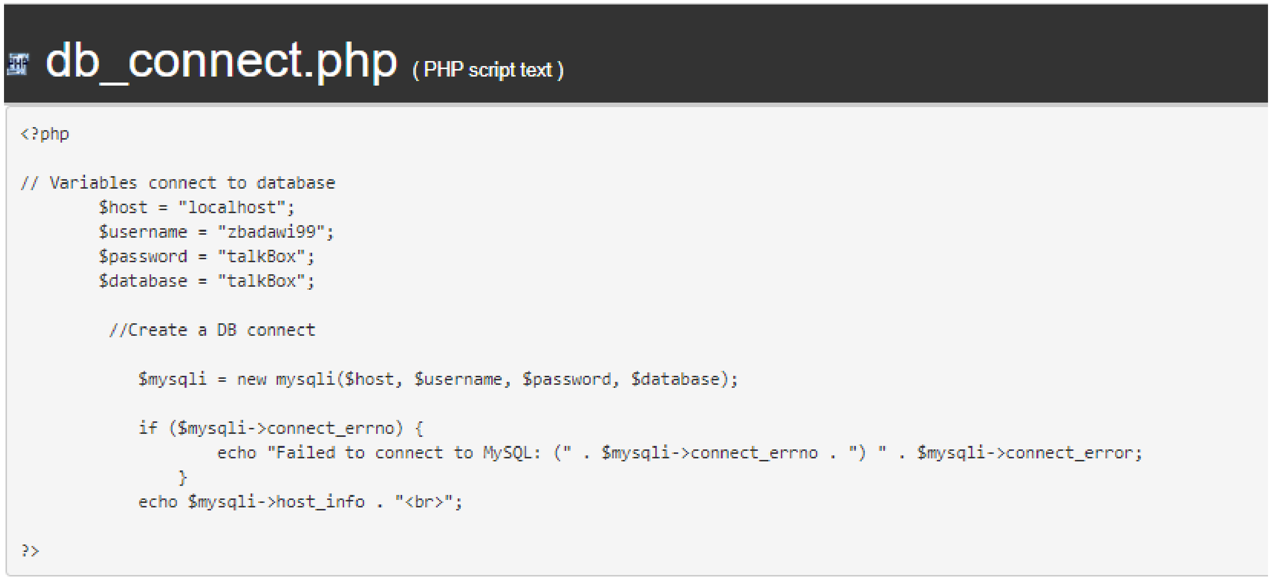
Emmet. “How to Install PHPMyAdmin for the Raspberry Pi.” *Pi My Life Up*, 11 Feb. 2021, pimylifeup.com/raspberry-pi-phpmyadmin/.

Hussain, Aamir. “Raspberry Pi 3 Complete Tutorial - Let's Get Started.” *Trick i Know*, 20 Apr. 2020, trickiknow.com/raspberry-pi-3-complete-tutorial-2018-lets-get-started/.

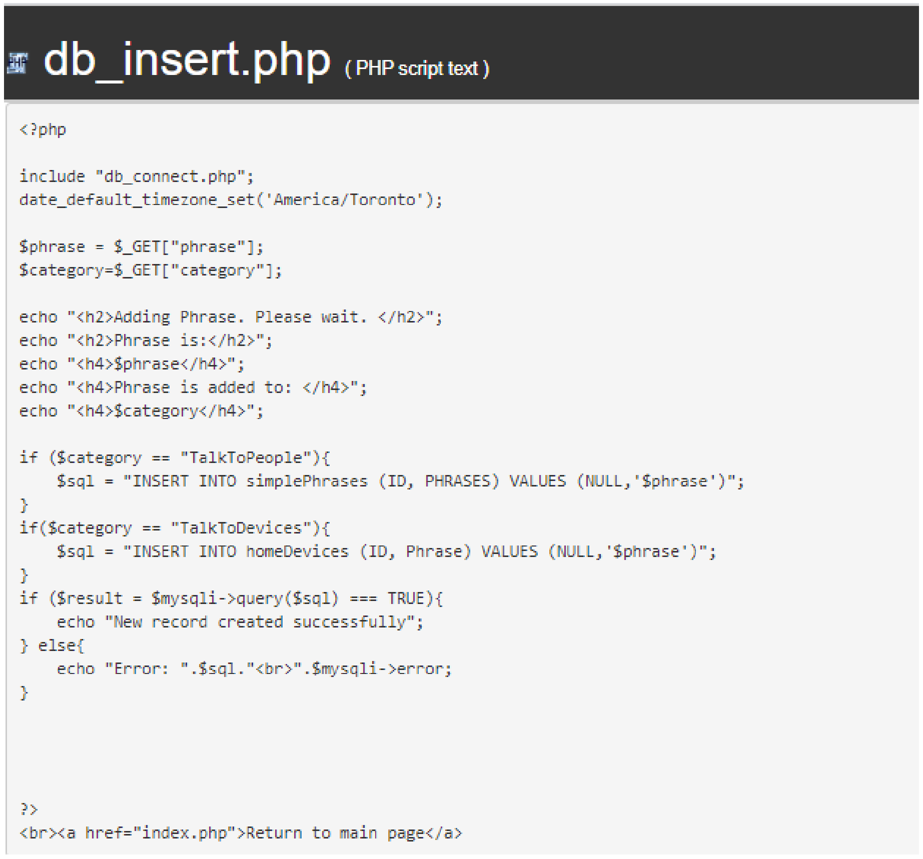
APPENDICES



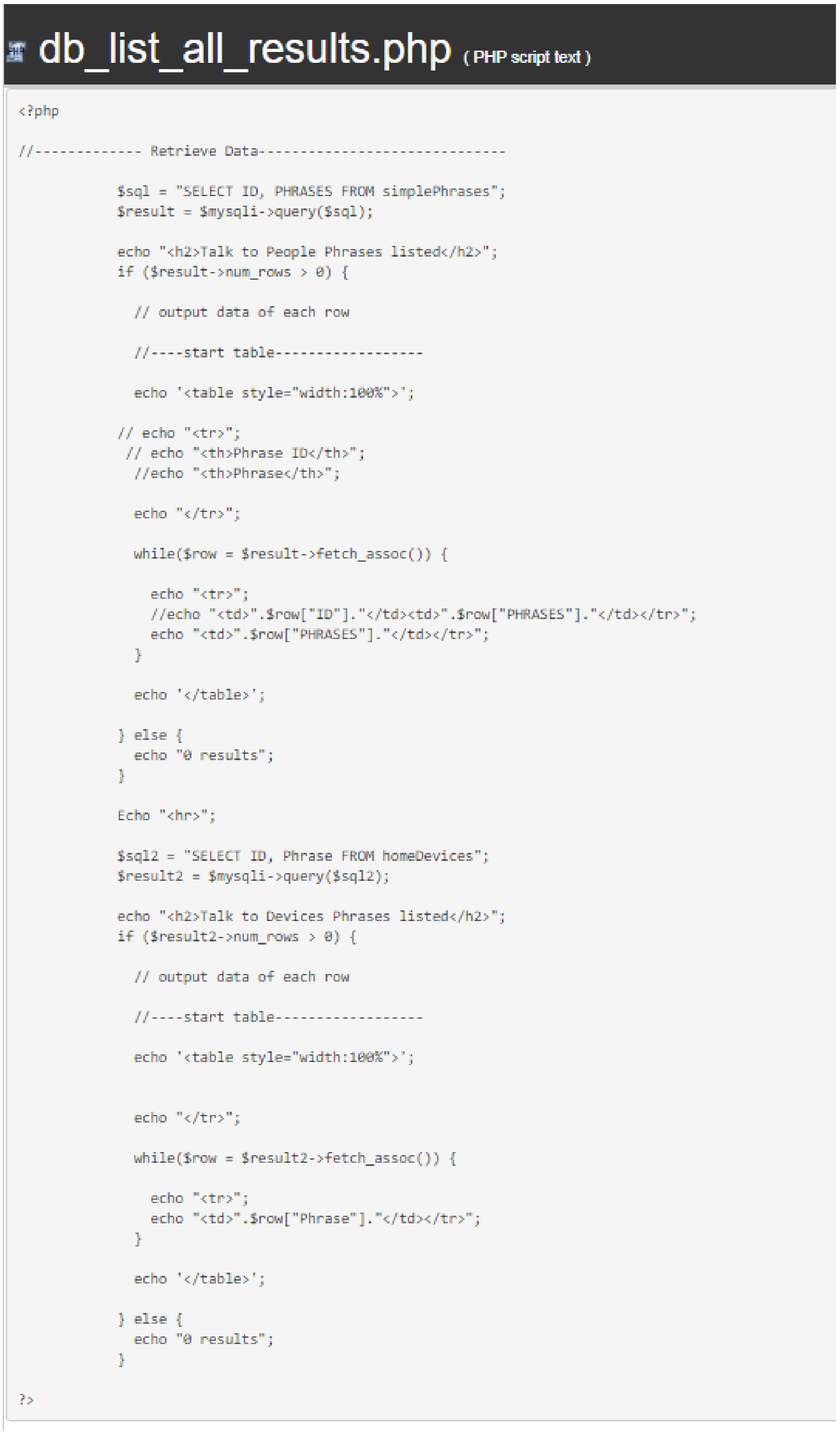
```
<html>
    <head>
         <?php

        include "db_connect.php";

   ?>

    </script>

    </head>

    <body>
        <h1>
            Phrases Entry Page
        </h1>



   <div id="map"></div>

    <hr>
        <h2>Enter Phrase Information</h2>

         <form action="db_insert.php">
             <!--------------- LAT-------------------->
            <label for="1"> Phrase:</label>
            <input type="text" id="phrase" name="phrase" value="enter a phrase">
            <select name="category" id="category">
            <option value="0" selected="selected">Select the phrase category</option>
            <option value="TalkToPeople">Talk to People</option>
            <option value="TalkToDevices">Talk to Devices</option>
            </select>
            <input type="submit" value="Submit">



        </form>
        <hr>






      <?php

         include "db_list_all_results.php";

            $mysqli->close();

        ?>

    </body>

</html>
```


db_connect.php (PHP script text)

```
<?php

// Variables connect to database
        $host = "localhost";
        $username = "zbadawi99";
        $password = "talkBox";
        $database = "talkBox";

         //Create a DB connect

            $mysqli = new mysqli($host, $username, $password, $database);

            if ($mysqli->connect_errno) {
                    echo "Failed to connect to MySQL: (" . $mysqli->connect_errno . ") " . $mysqli->connect_error;
                }
            echo $mysqli->host_info . "<br>";

?>
```


db_insert.php (PHP script text)

```
<?php

include "db_connect.php";
date_default_timezone_set('America/Toronto');

$phrase = $_GET["phrase"];
$category=$_GET["category"];

echo "<h2>Adding Phrase. Please wait. </h2>";
echo "<h2>Phrase is:</h2>";
echo "<h4>$phrase</h4>";
echo "<h4>Phrase is added to: </h4>";
echo "<h4>$category</h4>";

if ($category == "TalkToPeople"){
    $sql = "INSERT INTO simplePhrases (ID, PHRASES) VALUES (NULL,'$phrase')";
}
if($category == "TalkToDevices"){
    $sql = "INSERT INTO homeDevices (ID, Phrase) VALUES (NULL,'$phrase')";
}
if ($result = $mysqli->query($sql) === TRUE){
    echo "New record created successfully";
} else{
    echo "Error: ".$sql."<br>".$mysqli->error;
}




?>
<br><a href="index.php">Return to main page</a>
```


db_list_all_results.php (PHP script text)

```
<?php

//------------- Retrieve Data-----------------------------

            $sql = "SELECT ID, PHRASES FROM simplePhrases";
            $result = $mysqli->query($sql);

            echo "<h2>Talk to People Phrases listed</h2>";
            if ($result->num_rows > 0) {

              // output data of each row

              //----start table-----------------

              echo '<table style="width:100%">';

            // echo "<tr>";
             // echo "<th>Phrase ID</th>";
              //echo "<th>Phrase</th>";

              echo "</tr>";

              while($row = $result->fetch_assoc()) {

                echo "<tr>";
                //echo "<td>".$row["ID"]."</td><td>".$row["PHRASES"]."</td></tr>";
                echo "<td>".$row["PHRASES"]."</td></tr>";
              }

              echo '</table>';

            } else {
              echo "0 results";
            }

            Echo "<hr>";

            $sql2 = "SELECT ID, Phrase FROM homeDevices";
            $result2 = $mysqli->query($sql2);

            echo "<h2>Talk to Devices Phrases listed</h2>";
            if ($result2->num_rows > 0) {

              // output data of each row

              //----start table-----------------

              echo '<table style="width:100%">';


              echo "</tr>";

              while($row = $result2->fetch_assoc()) {

                echo "<tr>";
                echo "<td>".$row["Phrase"]."</td></tr>";
              }

              echo '</table>';

            } else {
              echo "0 results";
            }

?>
```

db_list_all_results.php

```
<?php

//------------- Retrieve Data---------------------------

        $sql = "SELECT ID, PHRASES FROM simplePhrases";

        $result = $mysqli->query($sql);

        echo "<h2>Talk to People Phrases listed</h2>";

        if ($result->num_rows > 0) {


          // output data of each row

          //----start table------------------

          echo '<table style="width:100%">';

        // echo "<tr>";

         // echo "<th>Phrase ID</th>";

          //echo "<th>Phrase</th>";

          echo "</tr>";

            while($row = $result->fetch_assoc()) {

            echo "<tr>";

            //echo "<td>".$row["ID"]."</td><td>".$row["PHRASES"]."</td></tr>";

            echo "<td>".$row["PHRASES"]."</td></tr>";

          }

          echo '</table>';

        } else {

          echo "0 results";

        }


        Echo "<hr>";

        $sql2 = "SELECT ID, Phrase FROM homeDevices";

        $result2 = $mysqli->query($sql2);

        echo "<h2>Talk to Devices Phrases listed</h2>";
```

```
      if ($result2->num_rows > 0) {
        // output data of each row
        //----start table------------------
        echo '<table style="width:100%">';
        echo "</tr>";

        while($row = $result2->fetch_assoc()) {

          echo "<tr>";
          echo "<td>".$row["Phrase"]."</td></tr>";
        }
        echo '</table>';
      } else {
        echo "0 results";
      }
?>
```