

Satish Reddy Bethi
Shubham Desai
Zeru Zhang

HUSH 4.0: Endgame

Objectives

Remotely control the threshold.

Occupancy monitoring.

Creating a simple interface to show noisy, quiet areas and occupancy of the zones.

Overview

A real-life scenario.

The threshold values can be directly sent from a mobile phone or a computer at the front desk of the library.

Values for each table are displayed on an LCD screen for the user.

If the sound level goes above the threshold, a red LED blinks indicating that the person sitting around it is being noisy.

Furthermore, we used the ultrasonic sensor to detect occupancy of the tables.

All of the data is reflected on the UI which is available to students online.

What's New

New and simple UI to see quiet, noisy and vacant zones.

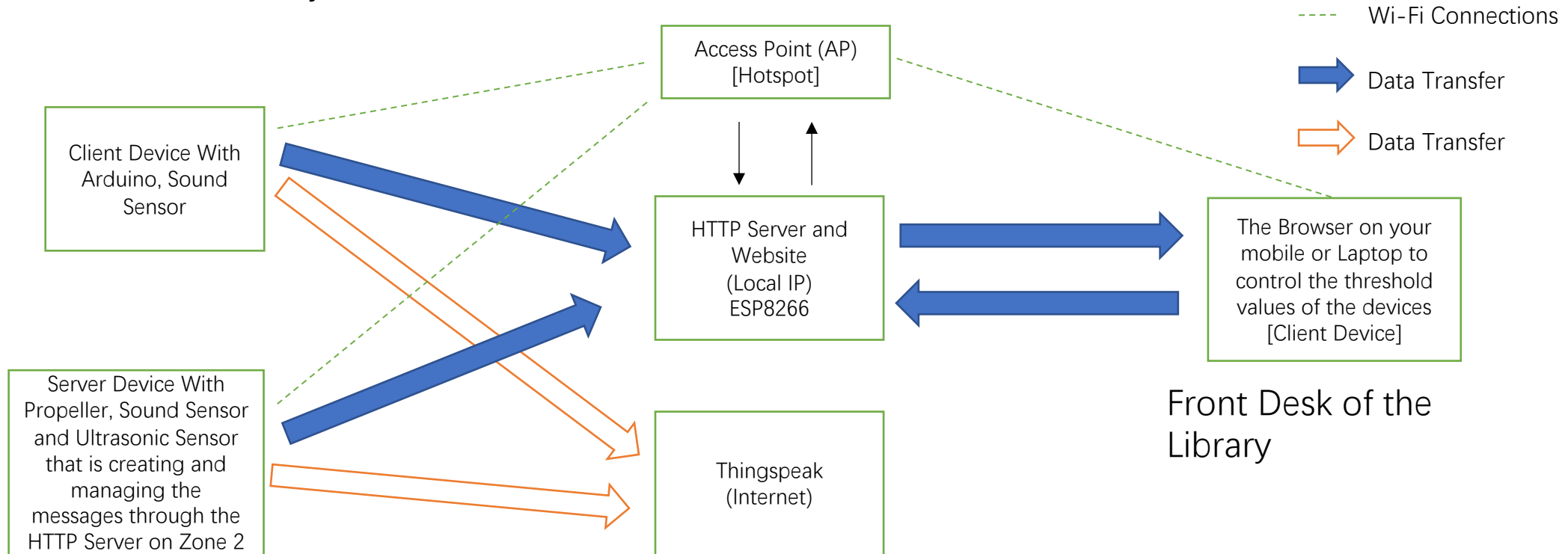
Occupancy detection.

Remote control of threshold.

Two-way communication.

Framework of the HTTP Server:

Zones in the Library



Circuits

Client device 1: Arduino with ESP8266 as the client device which receives data from the server, changes the device threshold, checks for occupancy and sends live data to ThingSpeak.

Client and Server device 2: WiFi module on the propeller maintains the local HTTP server while the propeller checks for occupancy and sends live data to ThingSpeak.

Client Device 3: The computer or any mobile phone will act as the third client device which can send the threshold values to the server.

Occupancy Detection

- Ultrasonic sensor on propeller board will be placed on the tabletop.
- Monitor the occupancy of the tables.
- Send data to ThingSpeak to display on the UI.

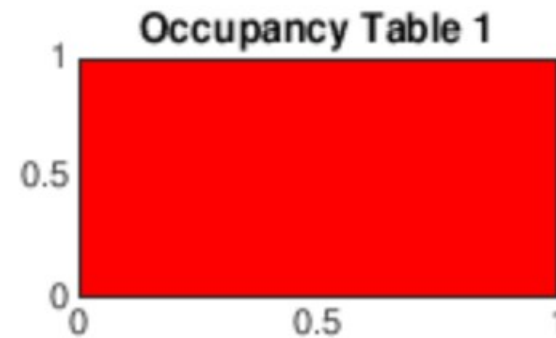
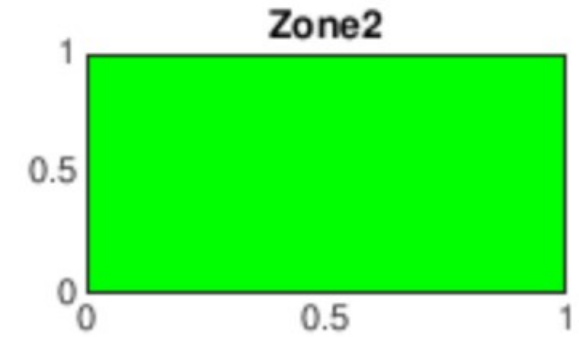
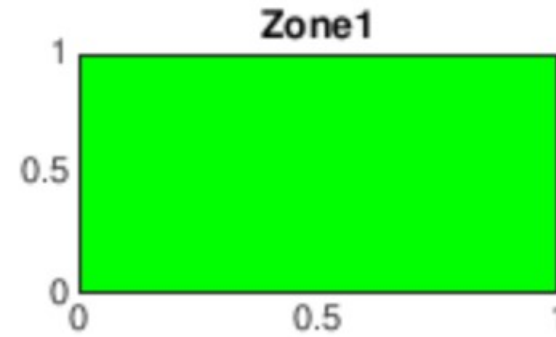


Remote Threshold Control

- Device that sets the threshold has to be connected to the same access point as other devices.
- Can control the threshold of all the devices at the same time.
- The front desk sends data to server and server sends it to other clients.

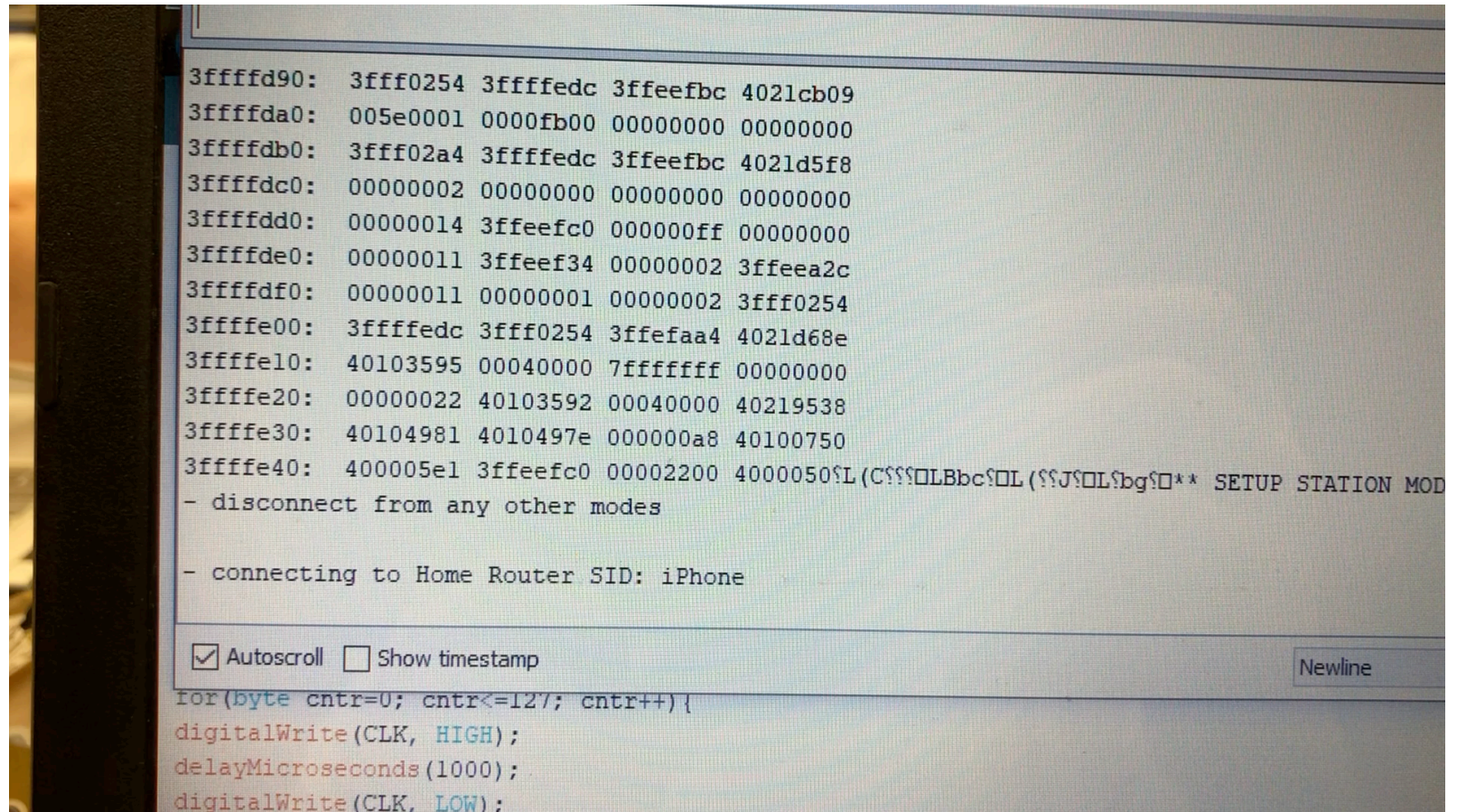
UI

- For Noise
 - **Green** – Quiet
 - **Yellow** – moderate noisy
 - **Red** – Very noisy
- For Occupancy
 - **Green** – Vacant
 - **Red** – Occupied



Challenges Faced

- Power fluctuations cause ESP8266 to reset.
- Due to this ESP8266 sends garbage values to the serial port.
- Debugging the ESP8266 was difficult after connecting them with microcontroller



```
3ffffd90: 3fff0254 3ffffedc 3fffeefbc 4021cb09
3ffffda0: 005e0001 0000fb00 00000000 00000000
3ffffdb0: 3fff02a4 3ffffedc 3fffeefbc 4021d5f8
3ffffdc0: 00000002 00000000 00000000 00000000
3ffffdd0: 00000014 3fffeefc0 000000ff 00000000
3ffffde0: 00000011 3fffeef34 00000002 3fffeea2c
3ffffdf0: 00000011 00000001 00000002 3fff0254
3ffffe00: 3ffffedc 3fff0254 3fffeefaa4 4021d68e
3ffffe10: 40103595 00040000 7fffffff 00000000
3ffffe20: 00000022 40103592 00040000 40219538
3ffffe30: 40104981 4010497e 000000a8 40100750
3ffffe40: 400005e1 3fffeefc0 00002200 40000505L(C??QLBbcQL(??JQL$bg$*** SETUP STATION MOD
- disconnect from any other modes
- connecting to Home Router SID: iPhone

☒ Autoscroll ☐ Show timestamp Newline
for(byte cnt=0; cnt<=127; cnt++){
digitalWrite(CLK, HIGH);
delayMicroseconds(1000);
digitalWrite(CLK, LOW);
```

Advantages

Improved the existing library occupancy detection system from wired to wireless.

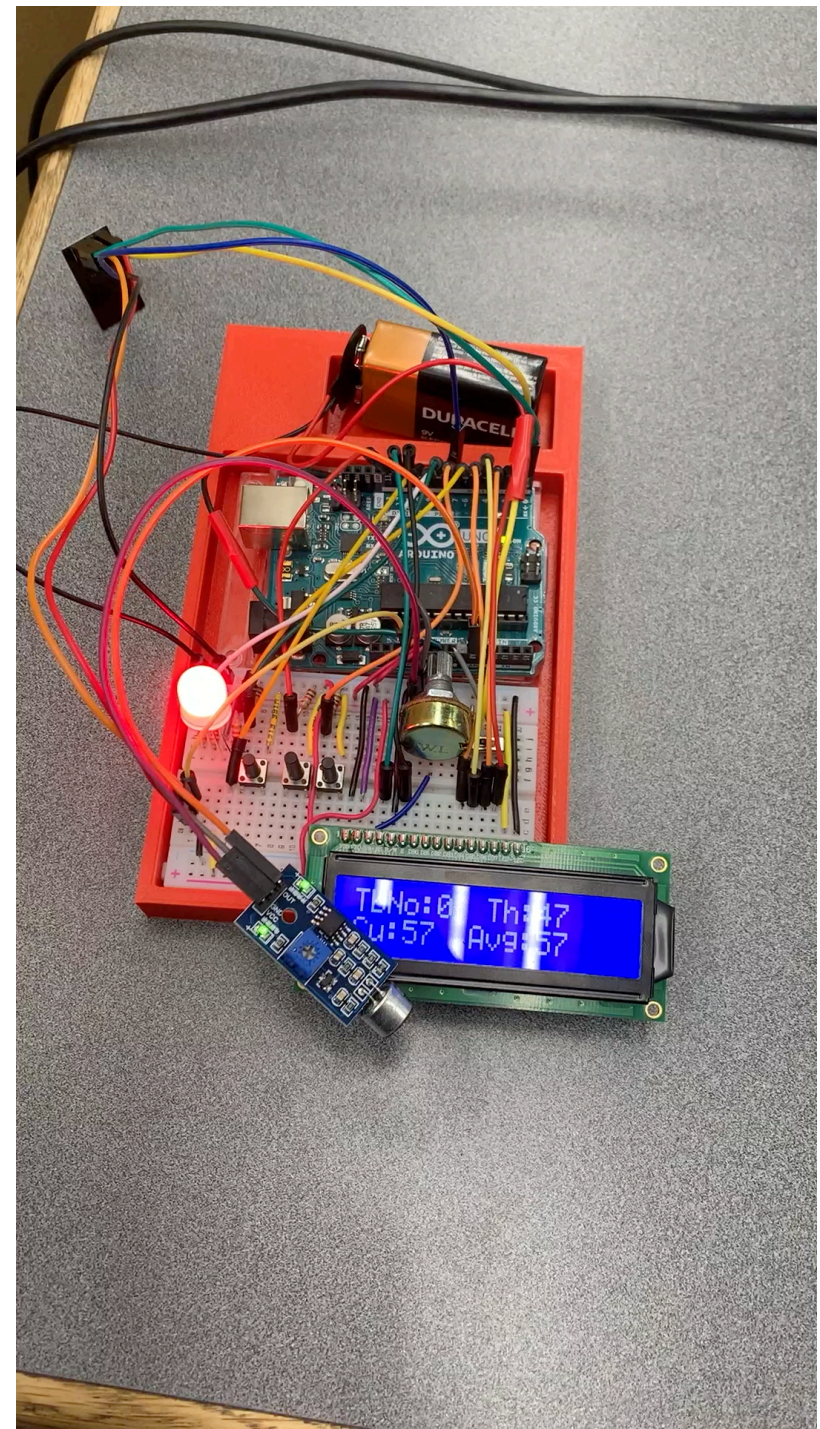
A perfect multifunction integral kit for each library table.

No longer required to tune all the devices separately.

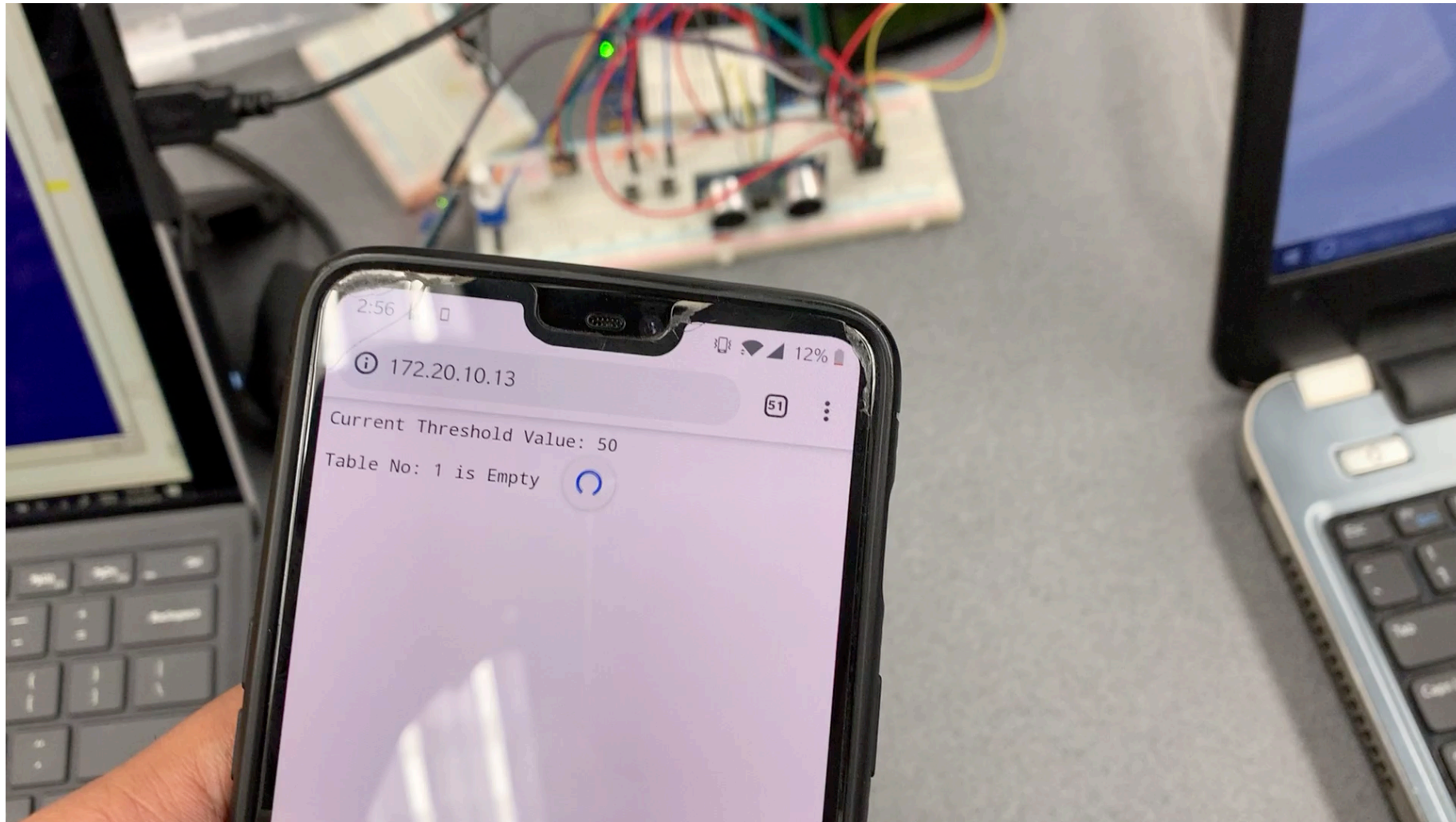
Made it much easier for librarian to manage all the tables.

More convenient for students to choose the study zone.

Demo for changing
the threshold wirelessly
from a mobile phone



Demo 1 for occupancy detection



Demo 2 for occupancy detection

