Logo%20Main%20200

**Boston University**

**Electrical & Computer Engineering**

**EC464 Capstone Senior Design Project**

User's Manual

NoiseHub

Submitted to

Michael Hirsch

8 St. Mary’s St

(617) 353-2811

mhirsch@bu.edu

by

Team 8

Team NoiseHub

Team Members

Allen Zou allenzou@bu.edu

Benjamin Brewer bbrewer1@bu.edu

Ibrahim Chand ichand@bu.edu

Alexander Prior aprior@bu.edu

Submitted: 04/18/2022

**NoiseHub User Manual**

#### 1 Table of Contents

1 Table of Contents 2

2 Executive Summary 3

3 Introduction 4

4 System Overview and Installation 5

4.1 Overview block diagram 5

4.2 User interface. 6

4.3 Physical description. 9

4.3 Installation, setup, and support 7

5 Operation of the Project 10

5.1 Operating Mode 1: Normal Operation 10

5.2 Operating Mode 2: Abnormal Operations 12

5.3 Safety Issues 12

6 Technical Background 14

7 Relevant Engineering Standards 17

8 Cost Breakdown 18

9 Appendices 19

9.1 Appendix A - Specifications 19

9.2 Appendix B – Team Information 19

# 

# 2 Executive Summary

The problem NoiseHub aims to solve is one all college students experience: finding a suitable study space on campus can be challenging and consume precious time, better spent studying. NoiseHub intends to improve efficiency in students' search for study spaces by providing accurate and real-time information on study spaces across campus. This information includes room temperature, noise levels, and an estimate of current occupancy.

As a final deliverable, the NoiseHub team will present a fully functioning sensor suite and companion app which is tailored to each individual user. Users will be able to specify room aspects they find important, and the suggested study spaces will be filtered based on the users preferences.

# 3 Introduction

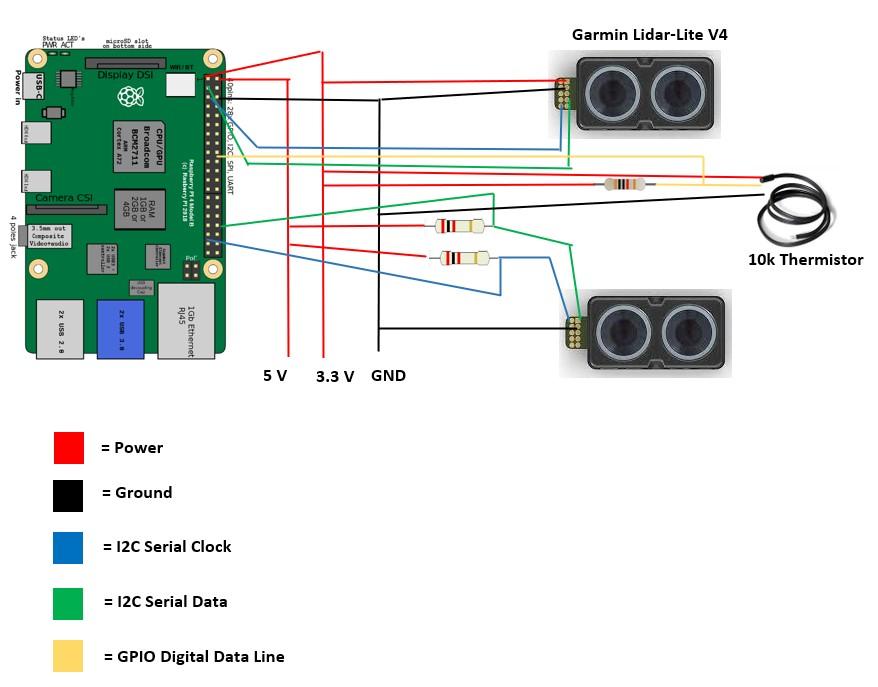
NoiseHub arose from a common issue among college students and people who use public work spaces: availability and conditions. Oftentimes students, researchers, faculty, and staff arrive at public works spaces only to find it too hot, too cold, too loud, overcrowded, awkwardly silent, etc. The three most important factors when searching for a study space are noise level, occupancy, and room temperature. Hence, NoiseHub aims to monitor, predict, and report these conditions to users in study spaces across campus. This process will be done with a fully integrated system consisting of sensor suites, backend data collection and analysis, and a mobile companion app for students looking for their ideal place to study (these students will be referred to as users). Organizations will be able to purchase this product and install it across their public spaces such as study spaces or cafes (these organizations will be referred to as clients).

There will be two types of wall mounted sensor suites. The first will contain a Raspberry Pi 4b (from now on referred to as Pi), and two Garmin LED Lidar-Lite v4s (from now on referred to as Lidars). The Lidars act as a tripwire in the doorway to measure occupants entering and exiting the room. This system doesn’t have complete accuracy, so user feedback will be incorporated as a failsafe. The second sensor suite will contain a Pi connected to an Adafruit Mini USB Microphone (from now on referred to as microphone) and a Programmable Resolution 1-Wire Digital Thermometer (from now on referred to as thermistor). The microphone captures sound and uses a state system based on each room's volume threshold to determine if the room is quiet (state 0), normal (state 1), or loud (state 2). The microphone records no audio and only measures RMS values to be processed for security. The thermistor measures temperature in Celsius which is then converted to Farenheit. Both Pis send the data they’ve collected to AWS services for data analysis and forwarding to the frontend.

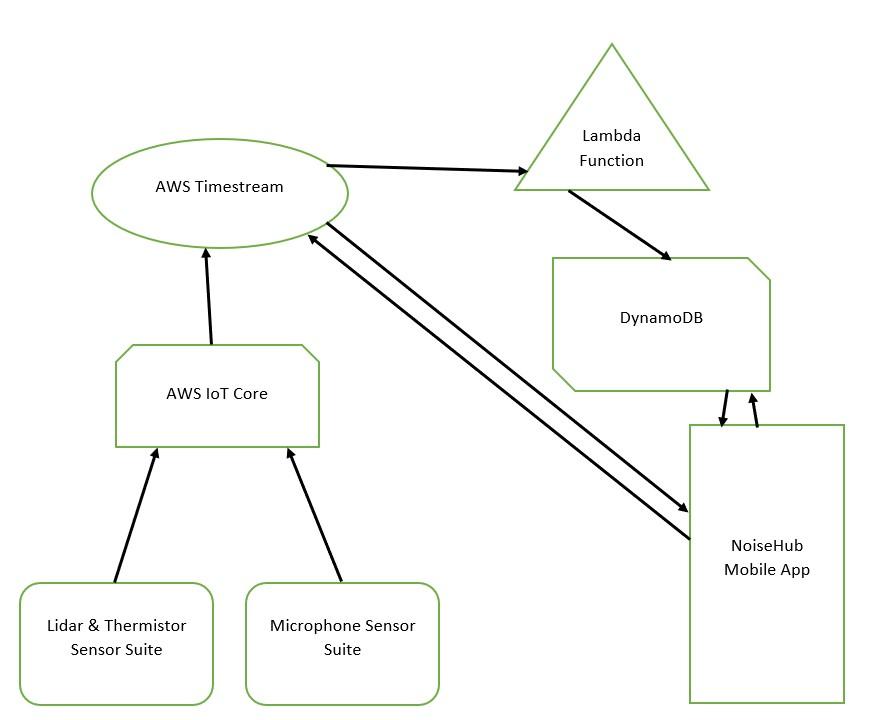
The AWS services provide easy authentication, secure data encryption, and powerful cloud computing. These services will handle user profiles as well as predicting peak noise, temperature, and occupancy times, working in conjunction with the companion app, from which users can query current room status, historic trends, and estimated conditions. The user will see all of this through an intuitive, clean UI/UX design.

# 4 System Overview and Installation

## 4.1 Overview block diagrams



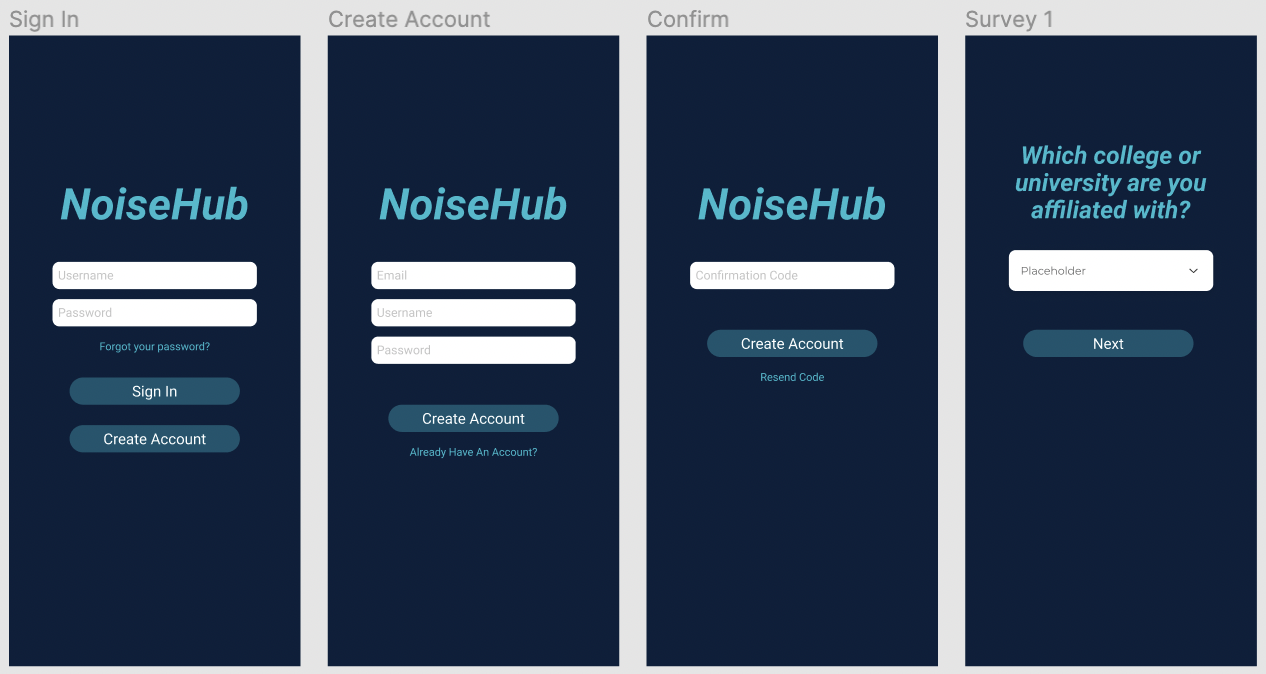
*Figure 4.1 Circuit diagram of Pi with Lidar and Thermistor*

**

*Figure 4.2 Overview block diagram of NoiseHub system*

## 4.2 User interface

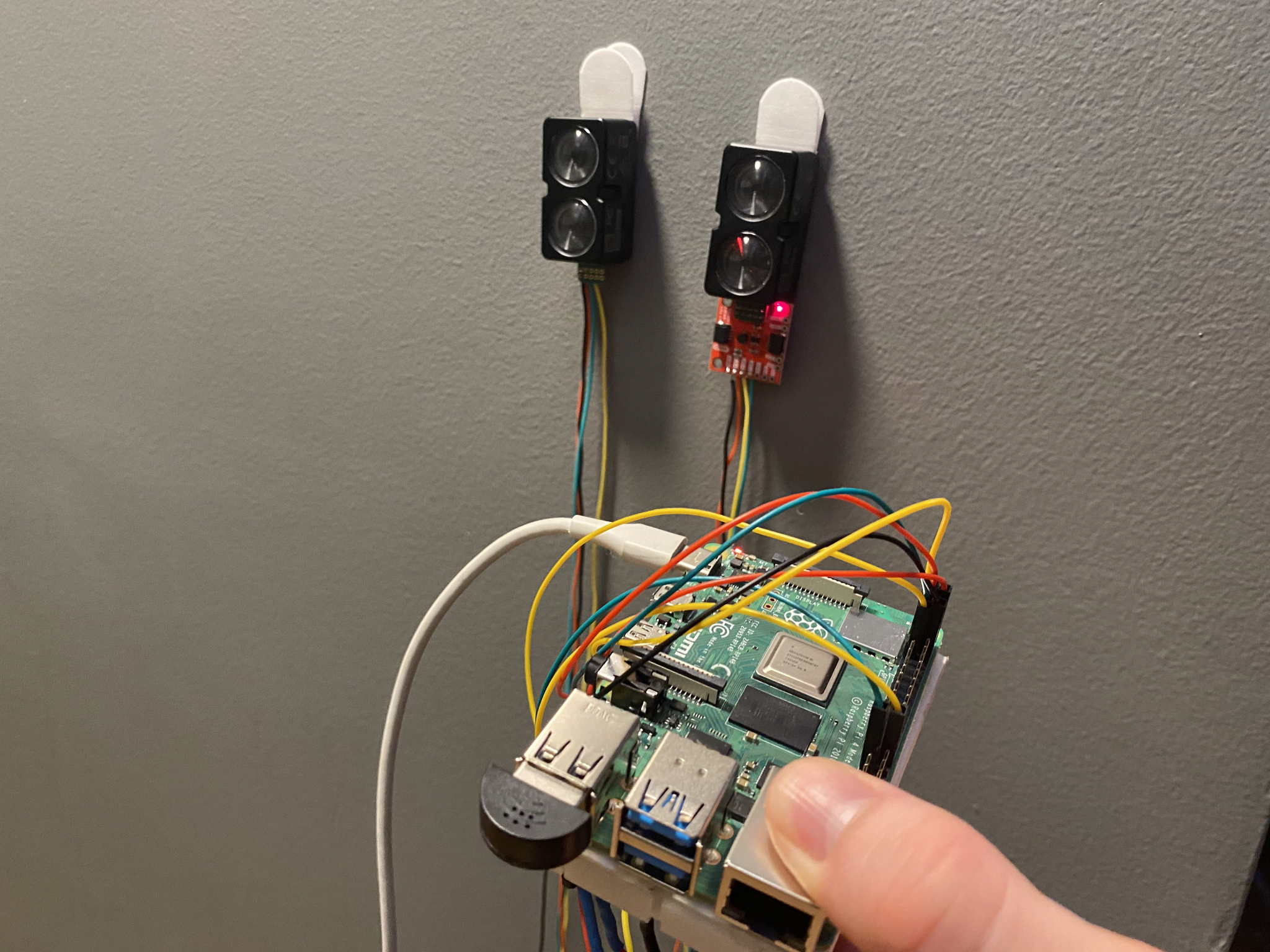
Noisehub developed a cross-platform mobile application that users can download to monitor the conditions of available study spaces based on the college or university with which they are affiliated. Users must create an account and answer questions regarding their affiliation and preferences. They can use the account to sign into the application and access information regarding available study spaces.



*Figures 4.3-4.10*

## 4.3 Physical Description

Figure 4.11 shows the mounting of the Lidars on the wall. The Lidar’s are spaced 3.5 inches apart from outside edge to outside edge. These can be mounted on a flat surface such as a wall, or on the inside of a door frame. The picture doesn’t show the case used to enclose the Pi or the case for the breadboard. Figure 4.12 shows a prototype microphone and thermistor Pi suite. The final product also has the breadboard encased, with the Pi and breadboard being mounted on the wall.



*Figure 4.11 The Lidar mounting*

## 

*Figure 4.12 Microphone and thermistor Pi suite*

## 4.4 Installation, setup, and support

When a client purchases the NoiseHub sensor suite, the NoiseHub team will install one Raspberry Pi near the room’s walkway and a power outlet. The soldered Lidar sensors will then be attached to the door frame in the correct orientation - entrance Lidar closest to the hinge and vice versa for the exit Lidar.

Four additional Pis will be configured. One Pi will act as an MQTT broker, receiving temperature and noise data from the other three Pis, which will be distributed around the room. These three “client” Pis will be sending data over MQTT to the “server” Pi every three seconds. The server Pi will average the noise and temperature values and forward the data to AWS every three minutes.

Then, using client-provided information about the study space, including address, space name, maximum capacity, and amenities, the Noisehub team will link sensors to their respective locations.

# 5 Operation of the Project

***5.1 Operating Mode 1: Normal Operation***

Mobile Application

1. **Sign In -** When the application is first opened, the user will be prompted to sign in using an existing account
   1. Enter account username (case insensitive)
   2. Enter account password (case sensitive)
   3. Click “Sign In”
2. **Create Account -** If the user does not have an existing account, the user has the option to create one which can be used in the future to sign in
   1. Click “Create Account”
   2. Enter email address
   3. Enter account username (case insensitive)
   4. Enter account password (case sensitive)
   5. Click “Create Account”
   6. Retrieve confirmation code from email sent to email address provided in (**2b)**
   7. Enter confirmation code
   8. Click “Confirm”
   9. Select affiliated college or university
   10. Click “Next”
   11. Select study space factor preferences
   12. Click “Next”
3. **Check Study Spaces -** Once a user signs in or creates an account, they will be taken to the *Home* screen
   1. **Search** - Users can search for study spaces by name by typing their query into the search box at the top of the *Home* screen
   2. **Preview** - The *Home* screen displays all available study spaces as individual cards that display current data regarding the conditions of the space: Study Space Name (top-left), Hours of operation (top-right), Noise Level (bottom-right), Headcount (bottom-middle), Temperature (bottom-right)
   3. **In-Depth** - Users can click on each study space card on the *Home* screen to view a dedicated page with historical data and amenity information
4. **Check In to a Study Space -** Each individual study space has a dedicated page from which a user can check in
   1. From the *Home* screen, click on the desired study space card
   2. Click the check mark in the top right corner
   3. Select the current level of business according to your best determination
   4. Click “Check In”

|  | **1. Sign In**  When the user starts the application, they will need to either sign in to an existing account or create one.  *Proceed to* ***6*** *if signing into an existing account*  *OR*  *Proceed to* ***2*** *if creating an account.* |  | **2. Create Account**  If the user selects to create an account, they will be brought to this screen where they must enter their email, username, and password. |
| --- | --- | --- | --- |
|  | **3. Confirm**  Once the user submits their information on the *Create Account* screen, they will receive an email with a confirmation code which they must enter here. |  | **4. Survey 1**  Once confirmed, a user must set up their profile. The first question prompts them to enter which college or university they are part of. |
|  | **5. Survey 2**  The second survey prompts a user to select their preferences for the conditions of a study space. |  | **6. Home**  Once a user signs in or completes registration, they will be brought to the *Home* screen. Here they can search for a study space by name, or select from the list to go to the *Study Space* screen. |
|  | **7. Study Space**  Each study space has an individual screen that provides in-depth data and information. |  | **8. Check In**  A user can check into a study space and provide information about how busy the space is to confirm accuracy of our data. |

*Figures 5.1-5.8*

## 5.2 Operating Mode 2: Abnormal Operations

Sensor Failure

1. Due to the high sample rate used on the lidar tripwire, occasionally the sensors will encounter an “I/O error.” This is just a software issue that causes the sensor readings to freeze up and requires a quick reinitialization. To do this, we have wrapped our headcount algorithm in a try, except block which is nested inside of a while true loop. As a result, every time we encounter this I/O error, no user intervention is needed and the sensor will start back up and continue counting headcount using the last saved occupancy value.

## 5.3 Safety Issues

1. Improper cable management of the Lidar sensor suite can create a tripping hazard. This could lead to injury of users, with the client at liability for improper maintenance.
2. Improperly mounted sensor suites are at risk of falling onto users. This could lead to injury of users.
3. Improper management of administrative credentials can pose a security threat. Exposed credentials can lead to a breach of the clients control over their NoiseHub system, and, in cases of poor online security practice, could lead to a breach of the client's network.
4. This product's backend is susceptible to any threats AWS services are susceptible to. A breach of the backend can lead to compromised personal user information.
5. Users are recommended to not use open containers of liquids near the sensor suites. Spilling of liquid on the sensor suites can cause damage to the sensors and Pi.
6. This product should be mounted only in controlled environments. Areas like gymnasiums with unpredictable moving objects can hit the sensor suite, causing wires to come loose, damage sensors, damage the Pi, or break the casing.

# 6 Technical Background

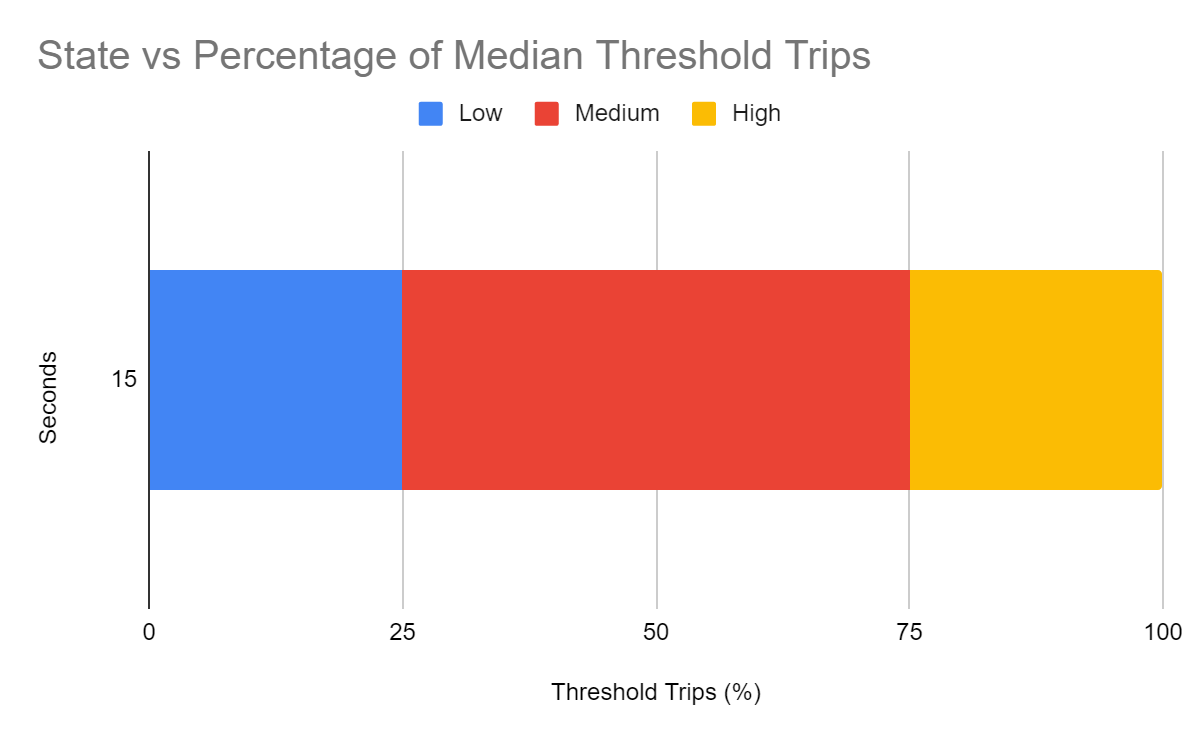
Upon installation of our NoiseHub sensor suite, the two Raspberry Pi systems will be linked to a study space and begin sampling data through every sensor.

Beginning with the room occupancy system, two Lidar units will be mounted on the door frame in parallel. They will be offset from one another by around 3 ½” and placed around average chest level. The two sensors will then constantly pull distance readings at a rate of once every 20ms. As a user walks past the sensors, their body will block the field of view for the Lidar and “trip” our threshold measurement. The Lidar sensors use a threshold of around 60% of the full door frame. Then, depending on the first sensor that reaches this defined threshold, the system determines if the movement is an entrance or exit. Due to the short distance between the sensors, there are certain cases where sensor trips can happen in the middle of an entrance/exit and be hard to distinguish. As a result, we implemented a secondary conditional to check if the expected sensor reads a shorter distance than the other. After a movement is detected, the system goes into a timeout state where it waits for a predetermined amount of time to account for most average walking speeds. After the timeout period, it begins sampling distance data again and waiting for the next threshold trip.

As the tripwire algorithm runs, a global headcount variable is incremented or decremented accordingly, and this number is then compared to the maximum room occupancy that is set by the client to determine a low, medium, or high room occupancy.

The microphone works by sampling RMS values every 0.5 seconds and comparing the measured values against a predefined threshold. This threshold varies from room to room and should reflect the median of the range of noise measured in that room. The RMS values are normalized on a [0,1] scale, but the threshold does not necessarily need to lie at 0.5.

After the proper threshold has been set, the microphone state system increases from low to medium (or medium to high) volume by measuring the number of “trips” above the threshold over a 15 second interval. If the system is in the low state and the number of trips is between 25% to 75% of the total possible trips over the 15 second interval, the state will increase from low to medium. If the system is in the medium state, and the number of trips is greater than 75%, the state will increase from medium to high. The same logic applies for decreasing states.



***Figure 6.1*** *Microphone State Change Visualization*

Lastly, our temperature sensor samples data upon request using the “One-Wire” communication protocol.

Every three seconds three Pis with both audio and temperature sensors will package and relay current data to a centralized pi. This is done to easily scale up the audio and temperature sensor systems to more accurately survey a room’s average data. To centralize data, the Pis all use mqtt to publish and subscribe to a topic on a mqtt broker which is hosted on the central Pi, which does not have any sensors or generate data.

The current room volume state, occupancy, and temperature is then sent to AWS Timestream every three minutes to ensure relevant information is displayed to end users on the mobile application.

The NoiseHub team is responsible for installing hardware in campus study spaces, including mounting the Lidars on door frames, wall mounting the Pis and ensuring it has access to wall power, and configuring the proper threshold for the microphone state system.

Once the hardware has been configured, all the user needs to do is download the mobile application available for both iOS and Android and create an account. Creating an account is as simple as providing an email and password and filling out a short user preferences survey. The survey will ask the user for their university and study space preferences, such as whether or not they care about temperature, noise, or crowdedness.

After creating an account and filling out the survey, the NoiseHub app will create a custom home page tailored to the user. The app will prioritize displaying study space information based on the user’s preferences. If the user cares only about room temperature and noise, it will prioritize displaying nearby study spaces with low room noise and a comfortable room temperature, ignoring crowdedness. The app will show the nearest study spaces that meet these criteria, offering a concise view of room name, hours, noise, crowdedness, and temperature.

Clicking into one of the study space cards will display room address, amenities, and a graph of historical room data. This data can be used by users to compare current room conditions and determine if a given room is busier or calmer than usual. Additionally, the app will use historical data to predict peak hours for crowdedness and noise in each study space for planning ahead.

# 7 Relevant Engineering Standards

**7.1 Mechanical**

1. The device casing must be smaller than 7” L, 5” W, 4” H including the Pi, decibel, and temperature sensors.
2. The Lidar sensors will need to be cable managed around a door frame nearby the Pi.
3. The device must be quieter than ambient room decibel readings to prevent interference with the audio sensors - around 30 decibels.

**7.2 Data Analysis**

1. Data models will be leveraged via AWS Lambda to predict when peak and minimum noise levels occur.
2. Daily noise, temperature, and headcount data will be accurately graphed over a 24-hour period.

**7.3 Maintenance**

1. The device will require little to no maintenance, drawing direct power from a wall outlet and is able to be managed remotely over SSH.
2. Data is not stored locally on the device, so it is not constrained by storage.

**7.4 Accuracy & Cost**

1. Headcount accuracy will be greater than 55%.
2. Temperature accuracy will be within 5 degrees Celsius.
3. The total component cost will not exceed $500.

**7.5 Software Design and Coding Standards**

1. The backend code is written in Python following PEP8 formatting standards.
2. The frontend code is written in React Native.

**7.6 Communication and Internet Protocols**

1. The on-premises devices communicate with AWS services via MQTT over HTTP.
2. The mobile application queries data from AWS Timestream and DynamoDB using SQL syntax over HTTP.

**7.7 Operational Environment**

1. The on-premises device is intended to be used in college or university study spaces with a range of capacities, noise levels, and temperatures.
2. The device must be discrete and able to operate with minimal maintenance.

**7.8 Privacy Requirements**

1. The device does not store any audio or personal data. All generated noise, temperature, and headcount metrics are anonymized.
2. Noise data is stored only as qualitative values (Low, medium, high noise).
3. Temperature data is stored as Celsius values.
4. Headcount metrics are stored only as qualitative values (Low, medium, high occupancy).

# 8 Cost Breakdown

| Project Costs for Production of Beta Version (Next Unit after Prototype) | | | | |
| --- | --- | --- | --- | --- |
| Item | Quantity | Description | Unit Cost | Extended Cost |
| 1 | 2 | Raspberry Pi 4 Model B | $55 | $110 |
| 2 | 1 | Thermistor | $9.95 | $9.95 |
| 3 | 2 | 2 x Garmin Lidar-Lite V4 w/ Breakout Board | $75 | $150 |
| 4 | 1 | Adafruit Mini USB Microphone | $5.95 | $5.95 |
| 5 | 2 | Raspberry Pi Casing | $11.99 | $23.98 |
| 6 | 1 | Breadboard Casing | $9.95 | $9.95 |
| **Beta Version-Total Cost** | | | | $309.83 |

The cost breakdown uses MSRP for Raspberry Pis. Due to supply chain issues, the cost of Raspberry Pis has drastically increased but are not reflected here.

# 9 Appendices

## 9.1 Appendix A - Specifications

| **Requirement** | **Value, Tolerance, Units** |
| --- | --- |
| System Dimensions | 7” L, 5” W, 4” H |
| Lidar Headcount Accuracy | 65% ± 10% |
| Temperature Accuracy | ± 5° |
| Audio Level Accuracy | (Relative measurement, Low-High) |
| System Cost | Maximum of $500 |

## 9.2 Appendix B – Team Information

Team NoiseHub’s members include Benjamin Brewer, Ibrahim Chand, Alex Prior, and Allen Zou. All four members are computer engineers who were inspired to pursue this project after encountering frustrations trying to find suitable study spaces on Boston University’s campus. This project posed challenges throughout all areas of the full stack, including hardware and software, backend development, front end development, and communications.

Benjamin Brewer is a graduating computer engineer with passions in hardware engineering and low-level programming. He hopes to build his career in the automation industry and will be starting as a Software Engineer at Teradyne this summer.

Ibrahim Chand is a graduating computer engineer with passions in cloud computing and data analysis. He hopes to work in the data field and will be stepping up to a full-time role as a Software Development Engineer at Amazon this summer.

Alex Prior is a graduating computer engineer with passions in cybersecurity and machine learning. He hopes to pursue his passions in the consulting industry and will be starting as a Security Analyst at Accenture this summer.

Allen Zou is a graduating computer engineer with passions in embedded systems and cloud computing. He hopes to pursue a career in space and defense and will be starting as a Software Engineer at Lockheed Martin this summer.