SafeAir

Prof. Rania Hussien

Hunter North, Brian Arnold, Ethan Sepa, Rafka Daou, Terry Jung and Melinda Tran

Dept. of Electrical and Computer Engineering, Box 352500
University of Washington Seattle, Washington 98195-2500 U.S.A

This report includes the following components:

1. Team, Roles and Responsibilities
2. Product Requirements Document
3. Realistic Constraints and Engineering Standards
4. System Requirements Document
5. Project Schedule
6. Project Resources
7. Outline of Experiments
8. Trial Designs
9. Experimental Outcomes
10. Impact and Consequences
11. Conclusions and Recommendations

Team, Roles and Responsibilities:

Researcher:	
Hunter North (*Hardware Lead*)	Air Quality Measurement, Covid Risk Calculation, IOS App, Database
Rafka Daou (*Software Lead*)	Air Quality Measurement, Covid Risk Calculation, IOS App, Database
Ethan Sepa (*IOS Lead*)	Air Quality Measurement, Covid Risk Calculation, IOS App, Database
Brian Arnold (*Raspberry Pi Lead*)	Customer Mask Detection, Dashboard, External Device Control with Alexa
Terry Jung (*Team Manager*)	Customer Temperature Check, Dashboard, External Device Control with Alexa
Melinda Tran (STM Lead)	Business Capacity Counter, External Device Control with Alexa

Product Requirements Document:

SafeAir is a hardware system targeted towards mitigating the spread of COVID-19 by automating tasks that need to be performed to assure that necessary precautions and guidelines are followed. These tasks include automatic checking of patron's temperature and patron mask compliance. In addition, the device is capable of tracking the building capacity, ventilation and air humidity levels which may affect the rate at which a disease spreads. A web server will be designed to stream data of present real time measurements about the air quality/ ventilation and building capacity which will be used in-app by managers and patrons of the business. In addition, a monitor will be displayed in the store so that customers can see real-time measurements without needing to access the app store.

The following figure(s) readily convey the process of how this system is designed to work.

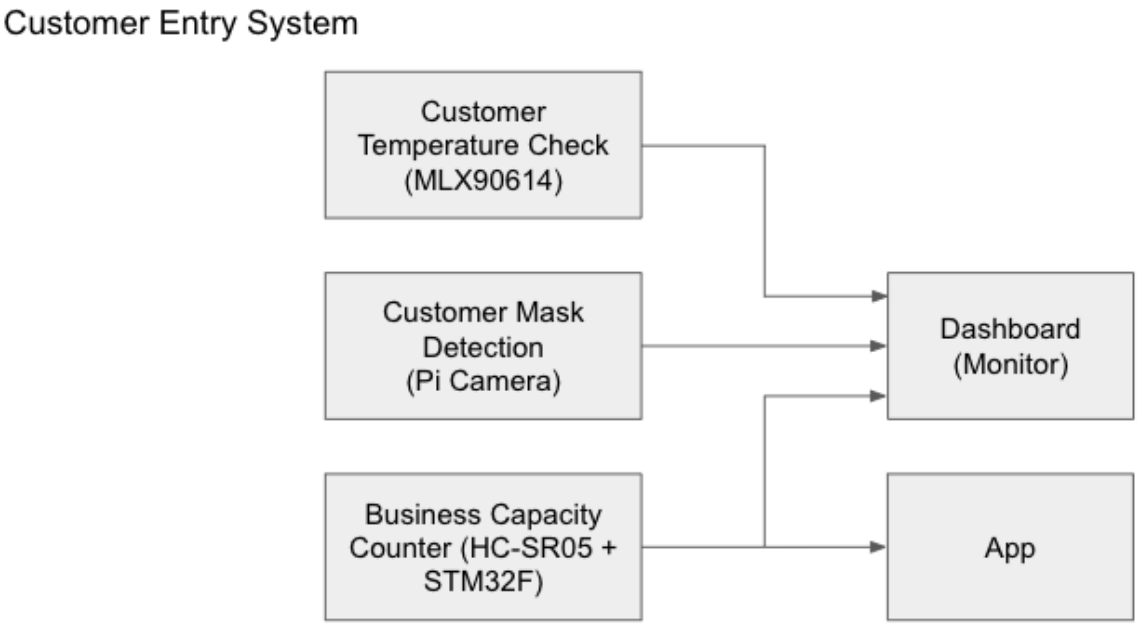


Figure 1: Customer Entry System hardware schematic

Figure 1 explains the hardware setup for a patron temperature and mask compliance system. Before entering the store, a customer stands in front of the camera and is checked to see if they are wearing a mask. If the individual is wearing a mask they still are not granted access to the store unless they have also passed the temperature check. Upon successfully completing both of these tasks, the individual is granted green access to be admitted into the business. The purpose of having a machine conduct the patrons temperature and mask compliance rather than an employee, is targeted towards eliminating the human to human interaction in the case that a customer is holding a virus.

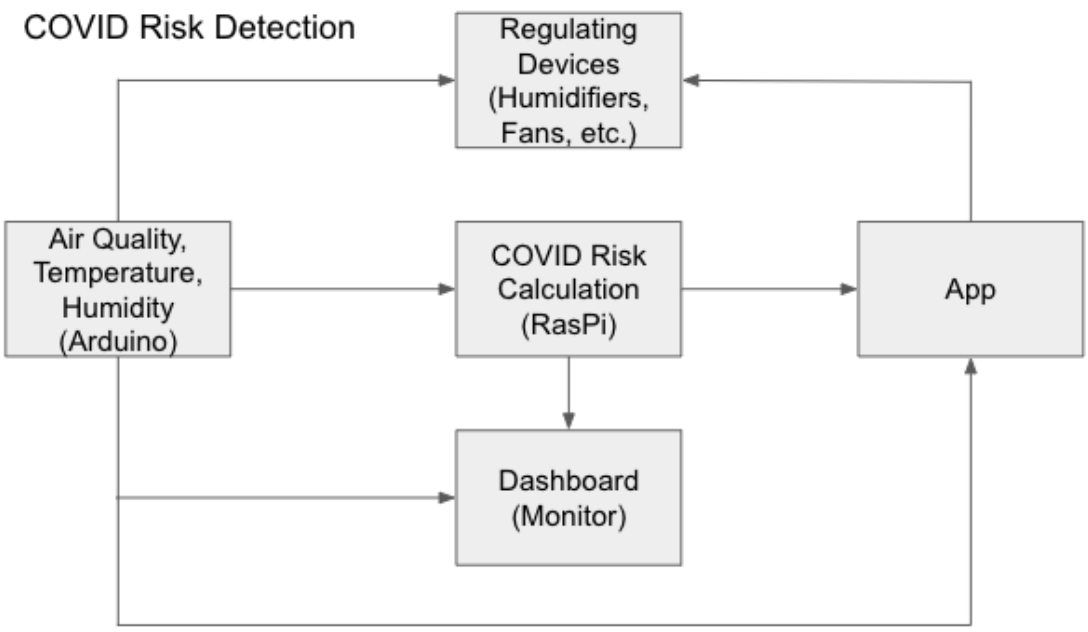


Figure 2: Air quality ventilation hardware schematic

Inside the business, a monitor displays the ventilation status of the business. The monitor is updated with real-time measurements of the temperature, humidity and CO2 levels of the business every five-ten seconds. The patrons also have access to an in-app. By downloading this app and entering the location ID of the business you are interested in visiting, you are able to see the real-time ventilation measurements displayed. This information is valuable to have in advance of entering the store to make the decision if you believe the ventilation of the store is safe for you.

Another important feature of the in-app is that business owners have the ability to set predefined setpoints and regulate external devices in the case that the ventilation reaches unsafe levels. The external devices include fans, heaters, dehumidifiers, etc.

The goal of our product, SafeAire is to develop a device that will slow the spread of COVID-19 and allow consumers to feel comfortable indoors thus allowing businesses to recover financially. SafeAir holds a strong market relevance as COVID-19 is an ongoing pandemic that is affecting businesses and individuals each day. The main purpose of this product is to foster an environment for businesses to protect themselves against the spread of COVID-19 and keep the economy moving forward. In addition, allowing individuals to feel safe while visiting stores, knowing that the businesses are taking necessary precautions against the virus.

Realistic Constraints and Engineering Standards:

Realistic constraints:

- The implementation of the regulatory devices were constrained to just sending commands to the Echo. The compatible devices were not purchased as a result of financial constraints. An Alexa compatible fan would cost over $120, nearly doubling our budget. This constraint was a rigid constraint because of the financial limitations of our budget.
- The accuracy of the object temperature sensor is constrained by the distance between it and the targeted customer. As a result, customers must approach 5 cm from the sensor for this design to work as intended. This constraint is rigid because the accuracy of the temperature sensor is important in detecting fevers for the customer entry system.
- The data from the climate is collected from its immediate surroundings. This constraint means that the sensor must be placed away from entrances and exits to accurately measure the climate of the business.

Engineering Standards:

Safety Standards:

Our project uses “low” voltages (less than 100 volts), which does not require licensing for residential or commercial work. Components of the systems are powered on less than 12 volts, reducing the risk of injury due to shock if a user were to misuse parts of the system. The system is also to be installed in non-wet locations to minimize water hazards. Wires and conductors from a circuit must be kept inside some kind of cable, raceway, or other protective agent. Specifics on these safety standards can be found in the NFPA 70, National Electric Code (NEC).

Our project also takes COVID-19 safety protocols into consideration. All hardware components of the system are hands-free, making the spread of viruses greatly reduced by not having to create a high-contact surface area to measure temperature or capacity. Users will also not have to touch the hardware to receive data from the system’s other sensors such as the humidity sensor, mask detector, and air quality sensor. Data is sent either to a user’s personal device or displayed on a public screen that is not meant to be touched. Making the components touch-free also reduces the risk of electric shock.

System Requirements Document:

1. Functional Requirements

    1.1 Main Node

    There is a central node in this IoT system that is used to drive the functionality across the entire system. This includes: loading a Losant Dashboard onto a webpage, capturing video, sensing temperature, identifying mask wearing, receiving data from other nodes via bluetooth, and pushing data onto databases.

    1.2 Climate Reader Node

    There is an additional node that serves the purpose of constantly reading data that represents the climate within the building. These characteristics include temperature, co2, and humidity. Additionally, this node will use bluetooth in order to send data wirelessly across our IoT system.

    1.3 Climate Response System

    The IoT system will autonomously manage a set of fans, dehumidifiers, humidifiers, and heaters in a way that will keep the climate within certain ranges. These appliances will turn on and off based on the data being read by the Climate Reader Node.

    1.4 Capacity Tracking Node

    The system will also feature capacity tracking nodes that can be placed at all doors leading to entrances and exits within the building. These nodes will use motion sensing to track how many people are inside the building. Additionally, this node will wireless send the capacity to the main node using bluetooth communication.

    1.5 Mobile Application

    The IoT system will also be accessible through an iOS application. This application will allow users to look into specific businesses to track previous and current climate quality. Users will also be able to alert store owners of unsafe conditions. Business owners will also have the option to determine acceptable climate conditions.

2. Size, Weight, and Cost Requirements

    2.1 Size Requirements

    All nodes in our system take up less than one cubic foot of volume. The only substantial unit of size is the monitor placed at the front of a business to display the entrance screen.

    2.2 Weight Requirements

    All nodes in our system weigh under 1 pound. The only unit of substantial weight would be the monitor placed at the front of a business to display the entrance screen.

    2.3 Cost Requirements

    All cost specifications can be found in the Project Resources section below.

3. Mechanical Requirements

    There are no mechanical requirements for this project.

4. Power Requirements

As far as specific power consumption constraints, there are no requirements. However, the monitor, main node, and each appliance will need to be plugged into an outlet to receive power. All other nodes will be powered by 5 volt batteries.

5. Thermal Requirements

There are no thermal requirements for this project.

6. Communication and Interface Requirements

6.1 I2C Communication Protocol

One popular method of interfacing a microcontroller to peripheral modules is I2C communication. More specifically, we will use it to correctly configure climate reading and human temperature reading. Here is an outline of the protocol:

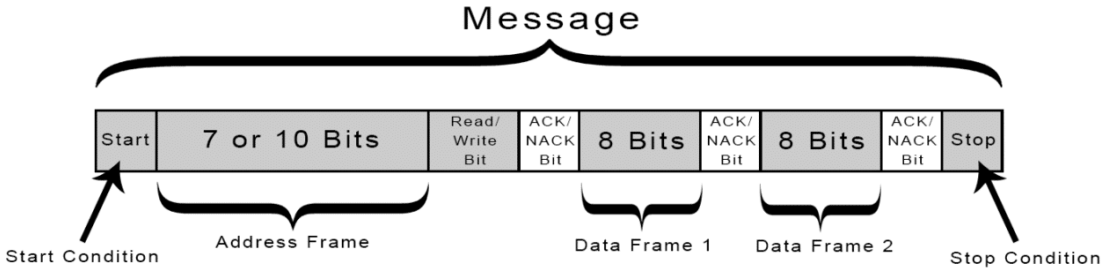


Figure 3: I2C communication protocol

6.2 Serial UART Communication Protocol and Bluetooth

Serial UART communication is also used in our system in order to send data to a bluetooth module. After our bluetooth module receives data, it sends the data wirelessly towards our main node -- which has a built-in bluetooth receiver. Here is a diagram explaining UART protocol:

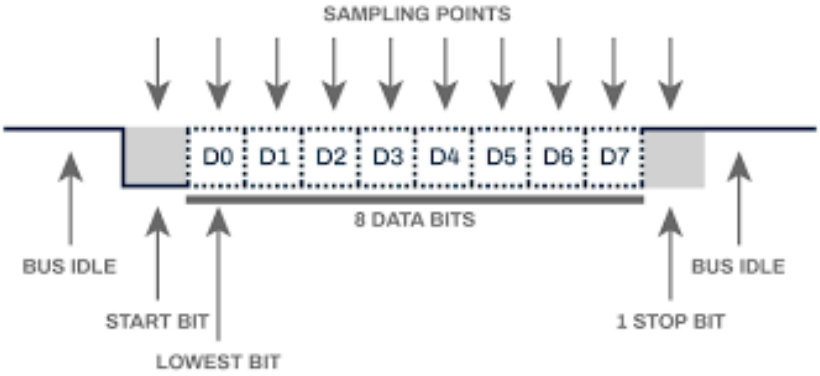


Figure 4: UART communication protocol

7. Control Requirements

There are no control requirements for this project.

8. Computation Requirements

8.1 Mask Tracking Algorithm

At the front entry system, we are displaying live video of the user trying to check in. On this video footage, we overlay max detection graphics. This creates a potential issue as

our mask detection algorithm must be computationally fast enough to keep up with the live video footage displayed to the user (or quality of UI will decrease drastically).

9. Software and Firmware Requirements

9.1 Programming Languages

Peripheral nodes will use the C programming language, the main node will use Python, and the iOS application will use Swift.

9.2 Repository

All code will be maintained and accessible at the following repository: https://github.com/hnorth99/SafeAir

10. Data Storage, Format, Security Requirements

10.1 Losant Dashboard

Our main node will constantly be uploading climate and capacity data through the Losant API. Losant will hold this data and allow us to present said data in a clean UI to a customer about to enter a business.

10.2 Firebase Database

Our main node will also be constantly adding data to a Firebase Realtime Database. This database will store all historical data in sorted order. The swift programming language compatible Firebase API is then used in the iOS application to present data to customers.

11. Precision and Accuracy Requirements

11.1 Climate Data Precision

When using the SCD-41, the carbon dioxide measurements can have an error up to 40 ppm, the ambient temperature can have an error of up to 0.8 degrees celsius, and the humidity can have error up to six percent.

11.2 Customer Temperature Reading

The MLX90614 peripheral module allows us to get accurate customer temperature readings with a margin of error of 0.5 degrees celsius.

12. User Interface Requirements

12.1 Customer Entrance User Interface

Before a user enters a store, we want them to be able to view live data that can be used to represent the safety within the store. We will show the current temperature, humidity, and co2 concentration. Additionally, we will highlight when these measurements are safe or unsafe. We will also display the status of a user's check-in, where we can ensure they are wearing a mask and do not have a fever.

12.2 Mobile Application User Interface

The first part of the user interface for our mobile application will focus on easily allowing the user to input a store or business they are planning to go to . Similarly to the entrance user interface, the app will also display live metrics and the safety of these metrics to the user. Additionally, this app will also display charts that will show historical data regarding the safety of the store.

Project Schedule:

The following figure is a Gantt chart that illustrates our project schedule.

TASKS	TASK OWNER	STATUS	START	END	DAYS
Project Planning:					
Idenify the problem and solution of our project	Everyone	Completed	1/3/22	01/12/22	1
Recieve project approval from teaching staff	Everyone	Completed	1/12/22	01/12/22	1
Complete project proposal report	Everyone	Completed	1/12/22	01/16/22	5
Complete project proposal slide deck	Everyone	Completed	1/12/22	1/16/22	5
Purchase necessary equipment	Everyone	Completed	1/12/22	1/16/22	5
Phase 1:					
Phase 1 specifications	Everyone	Completed	1/18/22	1/31/22	14
Detect temperature cartbon dioxide and humidity levels in a given space	Ethan/Rafka	Completed	1/22/22	1/28/22	7
Classify and alert hazardous measurements	Ethan/Hunter/Rafka	Completed	1/28/22	1/29/22	2
Create central system to recieve all the data of the current enviro.	Hunter	Completed	1/28/22	1/30/22	3
Configure Pi camera on the Raspberry Pi	Brian/Melinda/Terry	Completed	1/26/22	1/30/22	5
Integrate an air quality alert system for all businesses	Ethan/Hunter/Rafka	Completed	1/29/22	1/30/22	2
Phase 2:					
Phase 2 specifications	Everyone	Completed	2/10/22	2/23/22	14
Build non-contact infrared thermometer	Terry	Completed	2/10/22	2/14/22	5
Capture data from the camera incrementally using OpenCV	Brian	Completed	2/10/22	2/14/22	5
Develop an overlaid GUI monitor to display body temp and mask reading	Brian/Melinda/Terry	Completed	2/15/22	2/17/22	3
Display if a customer can enter	Brian/Melinda/Terry	Completed	2/16/22	2/18	3
Build a capactiy counter that will keep track of the number of people in the building	Melinda	Completed	2/7/22	2/20/22	14
Setup database for storing live air quality measurements	Ethan/Hunter/Rafka	Completed	2/18/22	2/20/22	3
Setup login page for both customers and owners	Ethan/Hunter/Rafka	Completed	2/20/22	2/22/22	3
Phase 3:					
Communicate between database and IOS app	Ethan/Hunter/Rafka	Completed	2/24/22	2/25/22	2
Build application to store information accessible to all individuals	Ethan/Rafka	Completed	2/24/22	3/1/22	5
Store sensor readings, and room capcaity on the application	Ethan/Hunter/Rafka	Completed	3/1/22	3/1/22	1
Preset air quality measurements that will activate external devices	Ethan/Hunter/Rafka	Completed	3/2/22	3/2/22	1
Allow owners to upddate thresholds on app and update database	Ethan/Rafka	Completed	3/2/22	3/4/22	3
Allow customers to send complaintss through application	Ethan/Rafka	Completed	3/2/22	3/4/22	3
Communicate with Alexa device	Brian/Melinda/Terry	Completed	3/4/22	3/5/22	2
Control external devices to control air quality with Alexa	Brian/Hunter/Melinda/Terry	Completed	3/5/22	3/6/22	2
Final:					
Project Poster	Everyone	Completed	3/7/22	3/14/22	8
Project Video	Everyone	Completed	3/7/22	3/14/22	8
Final Project Report	Everyone	Completed	3/7/22	3/14/22	8

Figure 5: Project Gantt Chart

Project Resources:
The following table defines the materials purchased for this project and the total cost covered. As a group we have stayed within our budget throughout the duration of our project.

Item	Qty	Part/Model #	Price	Purchase Link
Carbon Dioxide, Temperature and Humidity Sensor	1	1597-101020952-ND	52.90	link
Infrared Temperature Sensor	1	MLX90614ESF-BAA-000-TU	19.25	link
Pi Camera	1	Raspberry Pi Camera Module V2.1	27.19	link
Bluetooth	1	HC-06	Free	Owned
Motion Sensors	1 (pack of 3)	HC-SR501	7.29	link
Main Node	1	Raspberry Pi 4	Free	Lab Kit
Climate Reader Node (Microcontroller)	1	Arduino Mega2560	Free	Owned
Capacity Counter Node (Microcontroller)	1	Stm32F-Discovery	Free	Lab Kit
Amazon Alexa	1	Amazon Echo Dot	Free	Owned
Amazon Smart Plug	3	Amazon Smart Plug	24.99	link
Total Price			131.62	

Figure 6: Resources Table

In addition to the supplies used for this project as a group as resources we have also designated a meeting space where we have met two or more times a week to discuss our progress and collaborate on efficient ways to complete assigned tasks. This space has allowed us to ask

questions, learn from one another and keep every team member up to date with individual progress.

Outline of Experiments:
Over the course of this project, our group took steps as a whole to determine the design of SafeAir. The steps can be summarized as the following:

1. Discuss the goals and purpose of SafeAir
2. Break down the development of SafeAir to multiple systems that meet these goals and purposes
3. Research approaches to implement these systems
    a. Includes a list of the hardware needed
4. Build the system
    a. Order hardware
    b. Program hardware

After these steps are completed, we evaluate the approached solution by revisiting the purposes and goals we set for the system. With each system, this evaluation is envisioned differently based on what the purpose was. For example, the customer entry system was evaluated on its ability to check customers for COVID guidelines. This evaluation started off by testing each individual feature of the system. The temperature sensor was tested by placing objects with various temperatures and comparing the outputted value with a commercial temperature sensor. The mask detection feature was tested by running multiple trials revealing masked and unmasked group members to the camera. Lastly, the capacity counter was tested by simulating customers entering and exiting the business by walking through the motion sensors. Each of these tests aimed to evaluate whether or not the implemented approach meets the original purpose and goals set for the system.

In the case where the implementation of the approach does not work or does not pass our evaluation, we revisit step 3 in our process. While reflecting on why the previous approach did not work, we discuss and research another approach to satisfy our goals for the system. An example of this occurred in our climate control system. While we originally planned to integrate external regulating devices with power relays, we realized we could possibly run into issues cutting into the power cord and connecting the power relays. As a result, we discussed other possible approaches that would lead to more favorable outcomes. We finally landed on approaching this implementation by sending commands to an Echo device that controlled the external devices. This approach still met the purpose of the system by controlling the regulating devices that can affect the climate in the business.

Trial Design:

The SafeAirDevice is composed of three main components: air quality ventilation system, entry compliance system and IOS application. The three components have been integrated with one another to create a centralized system. The following section documents the design developed for the experiments along with an explanation of how the design is intended to work.

Air Quality Ventilation System:

The objective of the air quality ventilation system is to conduct real-time measurements of the ambient temperature, humidity and carbon dioxide levels in a room.

The proposed hardware design for completing this task is as follows:

Node 1:

The main node for conducting the real-time measurements of the ambient air quality is displayed in Figure 5. Figure 5 shows the SCD41 sensor wired to the Arduino Mega2560 board. In this task the Arduino board is programmed to read the ambient temperature, carbon dioxide, and humidity levels in a room by I2C communication channels with the SCD41 sensor.

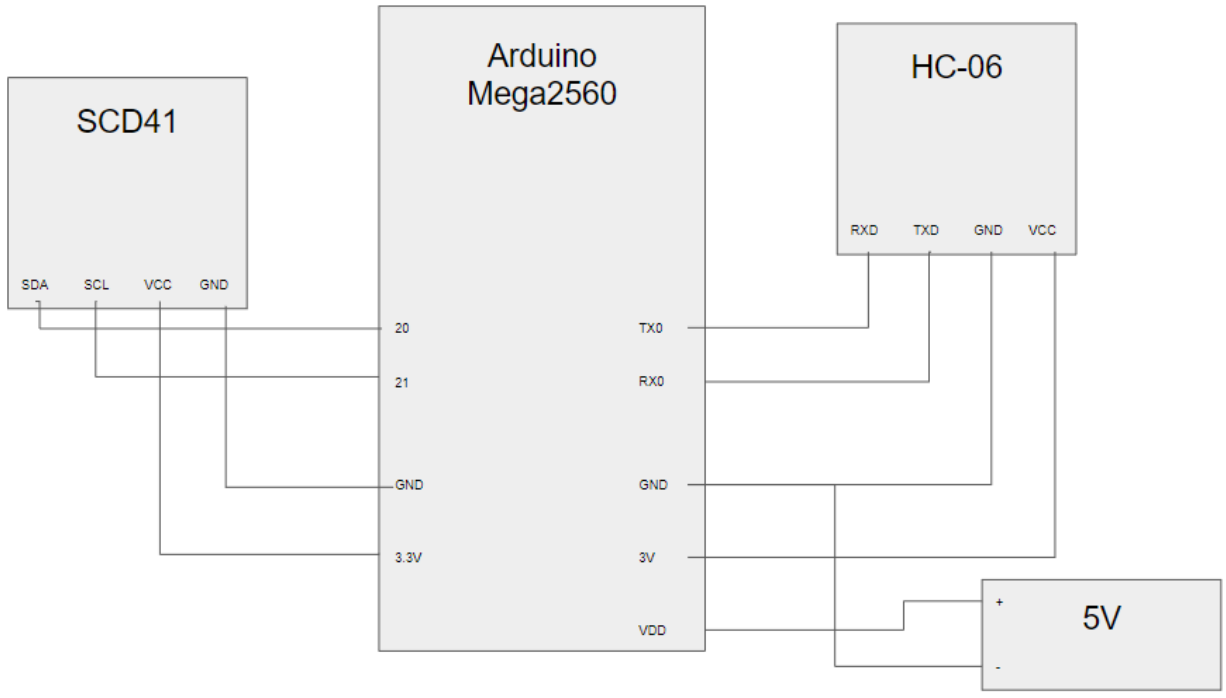


Figure 7: Air Quality Ventilation hardware schematic

The SCD41 sensor is wired to the Arduino Mega2560 board and the Arduino Mega2560 connected to a HC-06 bluetooth module. The Arduino board is programmed to read the ambient temperature, carbon dioxide, and humidity levels in a room by I2C communication channels with the SCD41 sensor. This information is then relayed wirelessly through bluetooth communication.

To conduct the air quality measurements, the SCD41 sensor is used as it features a quality humidity, temperature and carbon dioxide sensor that monitors measurements beneficial in demanding controlled ventilation, indoor air quality monitoring and carbon dioxide ventilation alarms. The sensor offers a 2.4V to 5.5V supply voltage range, fully calibrated I2C output and ±40 ppm + 5% accuracy reading. Figure 6 shows the layout of the functional block diagram for the SCD41 sensor.

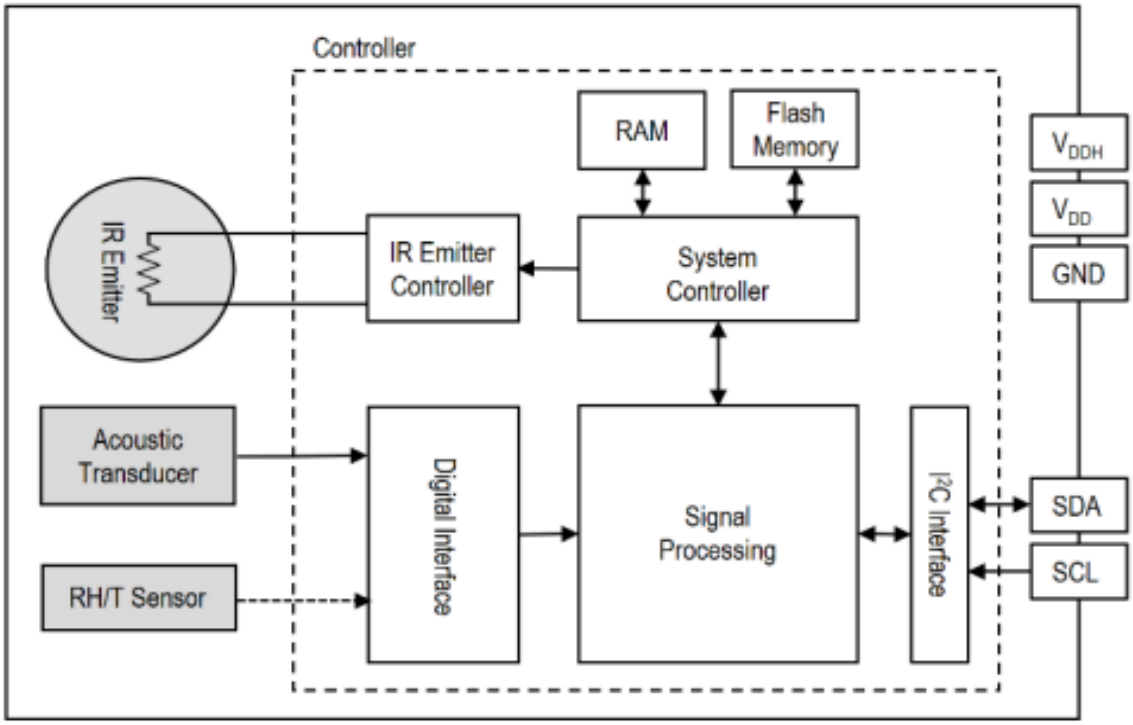


Figure 8: Functional block diagram for SCD41 sensor. [SCD41 Datasheet]

The SCD41 sensor has two power pins and two logic pins that are primarily used to connect with a microcontroller.

The following diagram shows the pin layout for the Arduino board and SCD41 sensor.

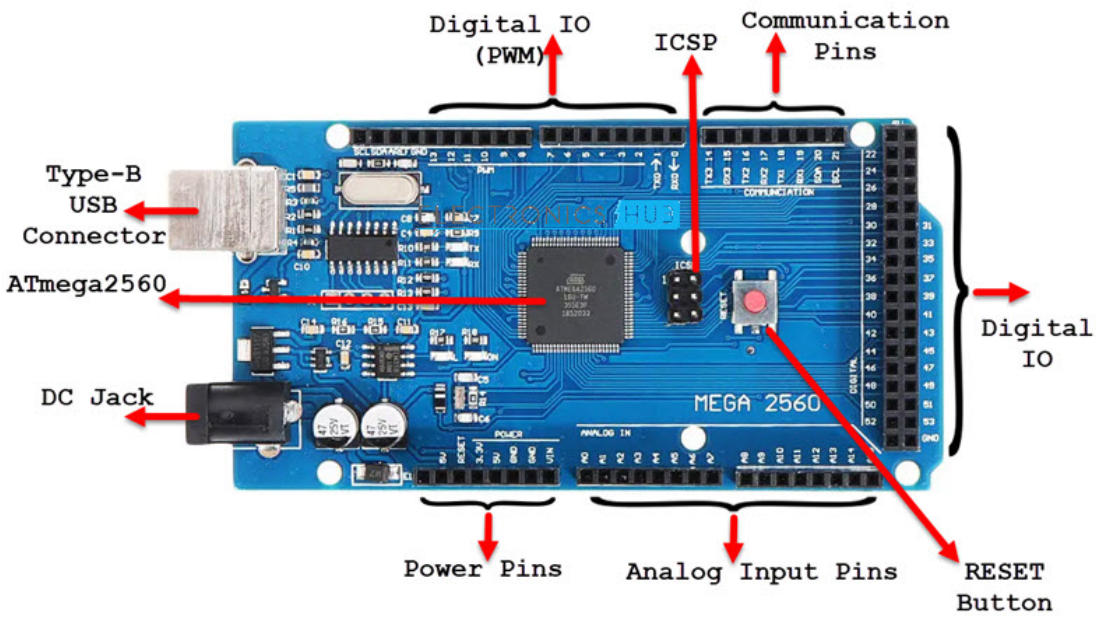


Figure 9: Arduino Mega Board layout

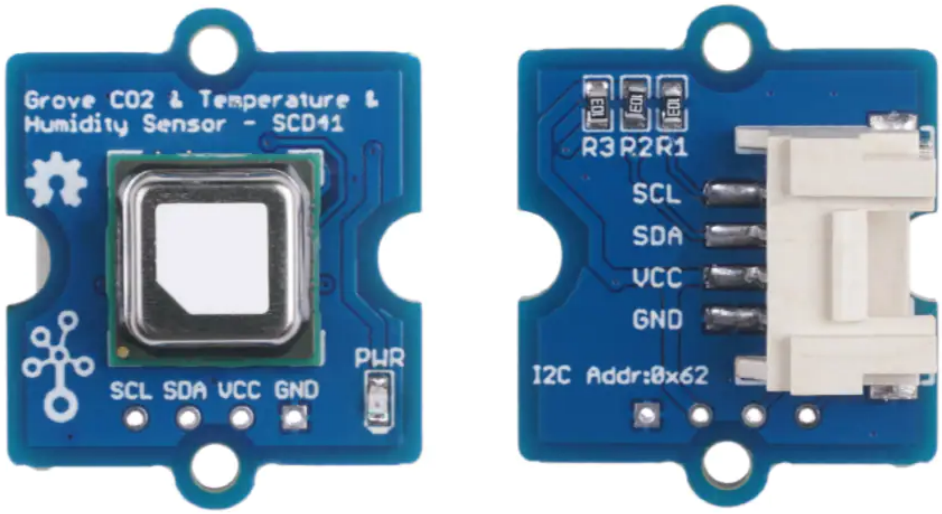


Figure 10: SCD41 sensor pin layout

The following connections are necessary to connect the SCD41 to the Arduino:

- VDD of the SCD41 to the 3.3V of the arduino board
- GND of the SCD41 to the GND of the arduino board
- SCL of the SCDC41 to the SCL of the arduino board
- SDA of the SCD41 to the SDA of the arduino board

The SCD41 sensor communicates with the Arduino through I2C communication. The SDA pin (I2C serial data) is used to transfer data to and from the sensor. The output bits are synchronized to the sampling of bits by the clock signal (SCL) that is shared among the master and slave device and are transferred bit by along along the SDA line.The SCL (I2C serial clock) pin on the SCD41 is used to synchronize the I2C communication between the master (microcontroller) and the slave (sensor).

The SCD41 sensor commands start_perioidic_measurement, read_measureemnt, and stop_perioidic_measurement were used to initiate measurement, check if current measurement is blocked and read data. The measurements recorded will also be uploaded to a server called firebase. Firebase will store the real-time data which will then be retrieved later on for the development of the ios application.

Node 2:

The second node of the air quality ventilation system is the ability to recognize when the real-time air quality measurements are unsafe. For testing purposes we first started by turning on remote LEDs that will be representative of devices that will respond to unsafe climate conditions.

Based on our research and understanding of unsafe climate conditions that impact the spread of viruses quickly, we have generated a flow chart that clearly identifies case by case what would happen if either the temperature, humidity or carbon dioxide levels are recorded unsafe.

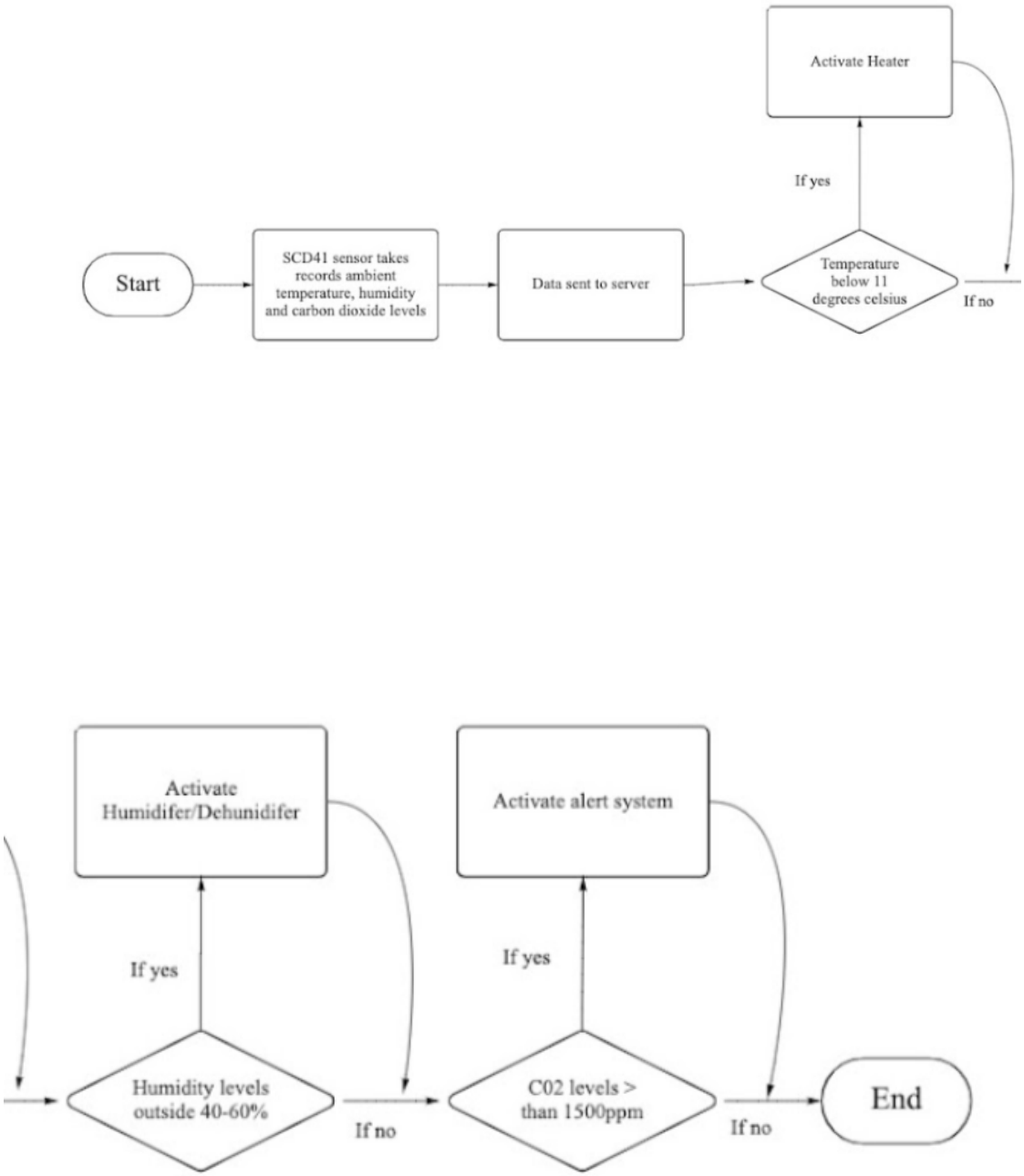


Figure 11: Air Quality Ventilation Flowchart

To develop this system, Figure 12 outlines the hardware for the system.

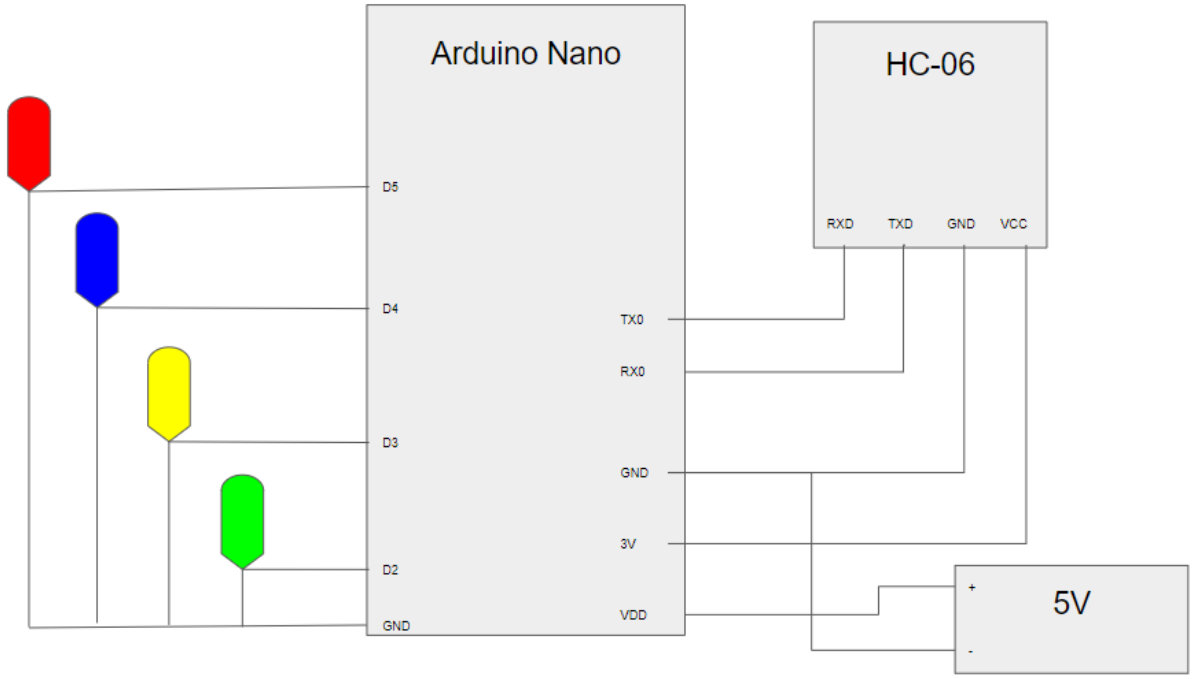


Figure 12: Hardware schematic for Alert System.

After successfully recognizing unsafe air quality conditions, we further expanded on this idea by being able to regulate external devices to improve the air quality conditions in the room. We have configured an Amazon Developer Account and installed alexa voice service on our Raspberry pi. We have utilized Alexa to step up routines. The routines are set to run when the air quality ventilation reaches unsafe measurements.

A fan, heater, humidifier, and dehumidifier are connected to an alexa smart plug in. When the routines are initialized and called the respected external devices will be regulated.

Entry Compliance System:

The second component of our device is an entry compliance system. The objective of this is to create a system that features a mask detector, temperature sensor, capacity counter and dashboard to display critical information to customers. The entry system is designed to mitigate the spread of diseases by only granting access to individuals who abide by business rules.

To better understand how the entry compliance system works, the following flowchart will guide you through cases.

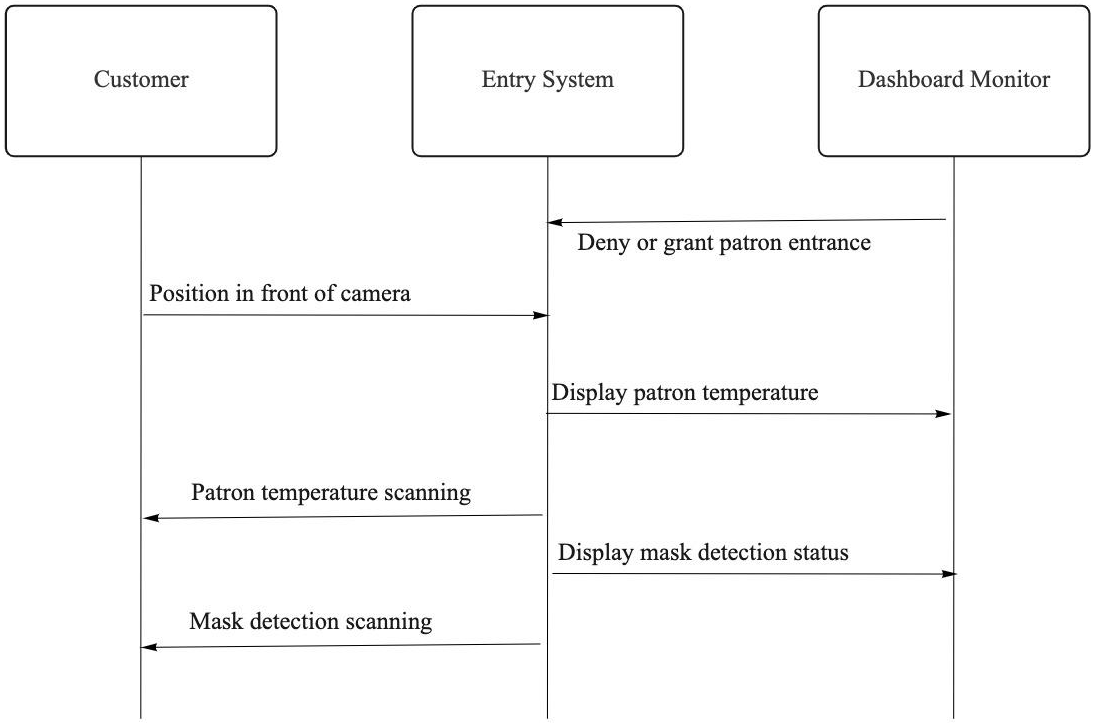


Figure 13: Entry System Sequence Chart

Raspberry Pi Node:
The first part of the entry compliance system is building a raspberry pi node that includes a mask detection system as well as the integration of a human temperature sensor. The following figure shows the hardware schematic for the raspberry pi node.

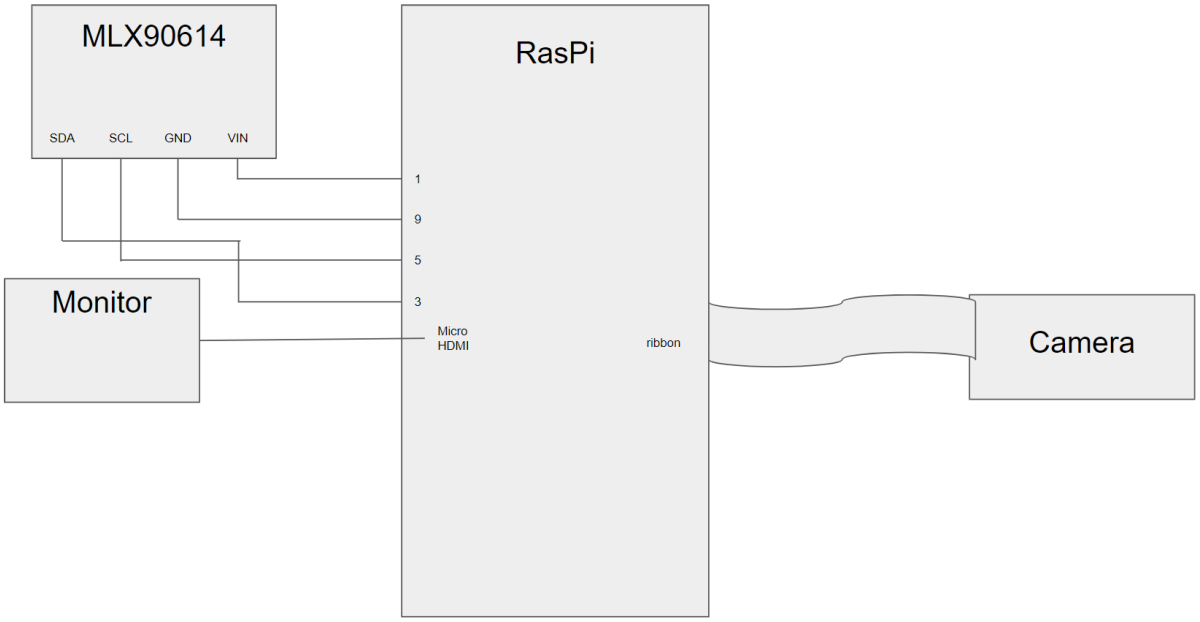


Figure 14: Raspberry Pi Node Hardware Schematic

In this schematic we have a Raspberry Pi Camera Module that will be used in tandem with the Raspberry Pi board to display video feed on a monitor and detect if the customer is wearing a mask as well as a MLX90614 is an infrared sensor to record human temperature.

The mask detection is performed on a video stream from the Pi Camera and displayed in an OpenCV. This incrementally captures images from the Raspberry Pi camera to produce a video feed. The processing images and predictions performed on the image overlay the GUI in the real time video feed. An overall entry message is displayed to indicate if the individual is wearing a mask and granted access or if no mask is detected and access to entrance is denied.

The next step that is checked to grant overall entrance is checking the patrons body temperature. The MLX90614 (infrared sensor) depicted in Figure 2. The block diagram for this sensor can be found in Figure 15.

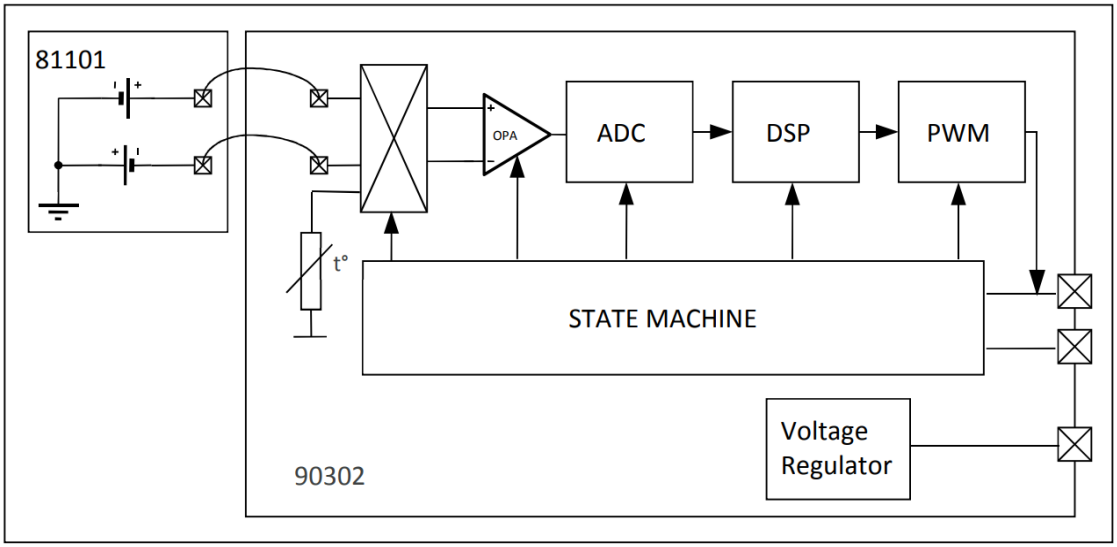


Figure 15: Block diagram of the MLX90614 sensor

This sensor has two modes of communication. The PWM mode and the I2C mode. The former method sends a digital on signal when the measured temperature reaches a set threshold value. The latter method relays the actual temperature value using the I2C protocol. In this lab, the latter method will be used so the actual value can be displayed to the customer on the dashboard.

In this lab, we will utilize the GPIO pins to connect to the I2C interface on the Raspberry Pi. The Raspberry Pi will be assigned as the master and the MLX90614 will be assigned as the slave. This will allow for the measurements from the MLX90614 to be received by the Raspberry Pi.

STM324F Node:
The STM32F407 is another microcontroller with features that specialize in audio processing on top of the general expected features of a microcontroller. For this lab, we will utilize the STM32F's GPIO pins as digital inputs connected to the HC-SR501 sensors.

The HC-SR501 sensor has 3 pins: GND, DOUT, and VCC. There will be 2 HC-SR501 sensors: an "inner" and an "outer." Both will be connected to a 5 volt battery in parallel. The connections between the HC-SR501 and the GPIO pins on the STM32F407 can be seen in Table 1.

Sensor Pin (inner sensor)	STM32F407 GPIO Pin
GND	Negative end of 5 volt battery
DOUT	Header2 GPIO40 (A15)
VCC	Positive end of 5 volt battery

Table 1: this table shows the connections between the inner HC-SR501 sensor and the STM32F407

Sensor Pin (outer sensor)	STM32F407 GPIO Pin
GND	Negative end of 5 volt battery
DOUT	Header2 GPIO42 (A13)
VCC	Positive end of 5 volt battery

Table 2: this table shows the connections between the outer HC-SR501 sensor and the STM32F407

The GPIO Pin numbers refer to the labels seen in Figure 16.

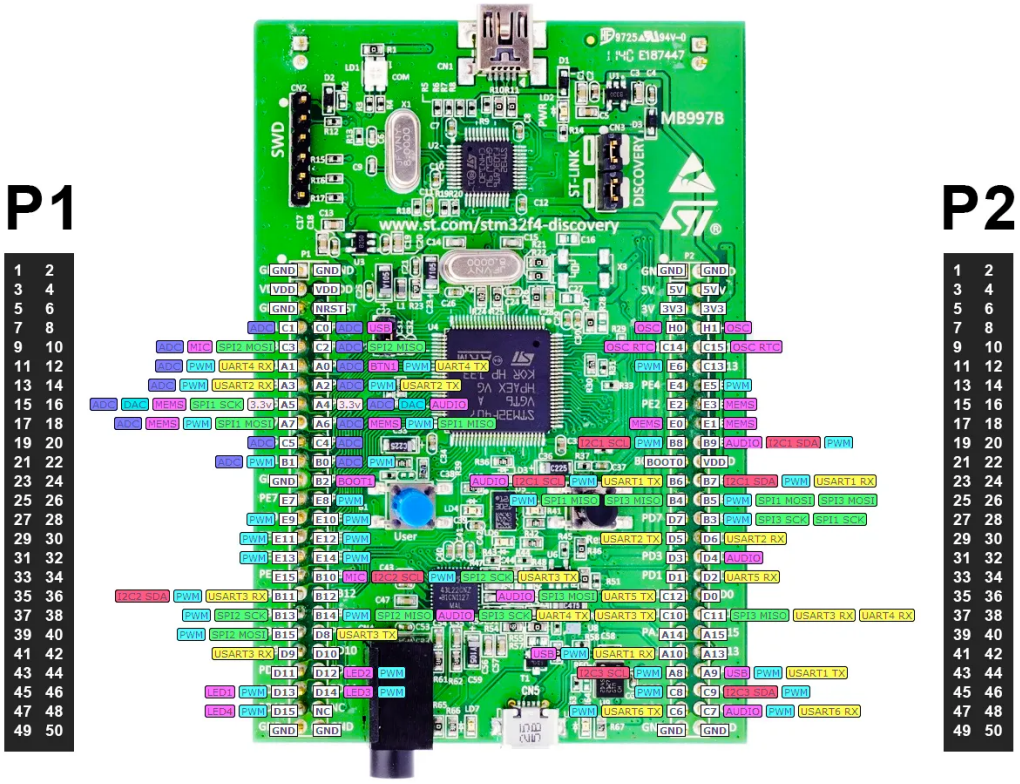


Figure 16: Diagram of the GPIO Pins and their number labels on the STM32F407

The HC-SR501 sensors will communicate with the STM32F407 through input/output pins, signaling high (3.3V) or low (0) into the device. This will be useful information when using the two sensors to calculate the capacity of a room. When installed in a frame, the outer sensor should face the outside of the room, and the inner sensor should face the inside of the room. Using input from the states of the sensors are high or low, increment or decrement a capacity variable accordingly. The following diagram of the finite state machine to calculate when someone has entered or exited the room.

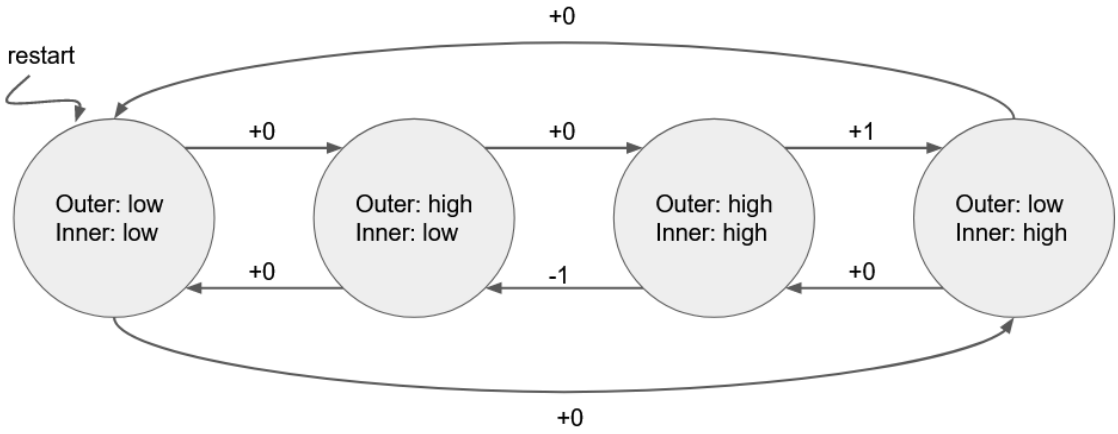


Figure 17: finite state machine diagram for the states of the 2 HC-SR501 sensors.

Depending on which sensor (outer or inner) turns low first from the state in which both sensors are high, +1 or -1 from the capacity. Otherwise, do not change the capacity. The importance of the HC-SR501 sensors is to keep track of the capacity in a given room. A setpoint will be determined by business officials to control the number of individuals in a room at a given time.

The following figure depicts the entry compliance system through a UML chart.

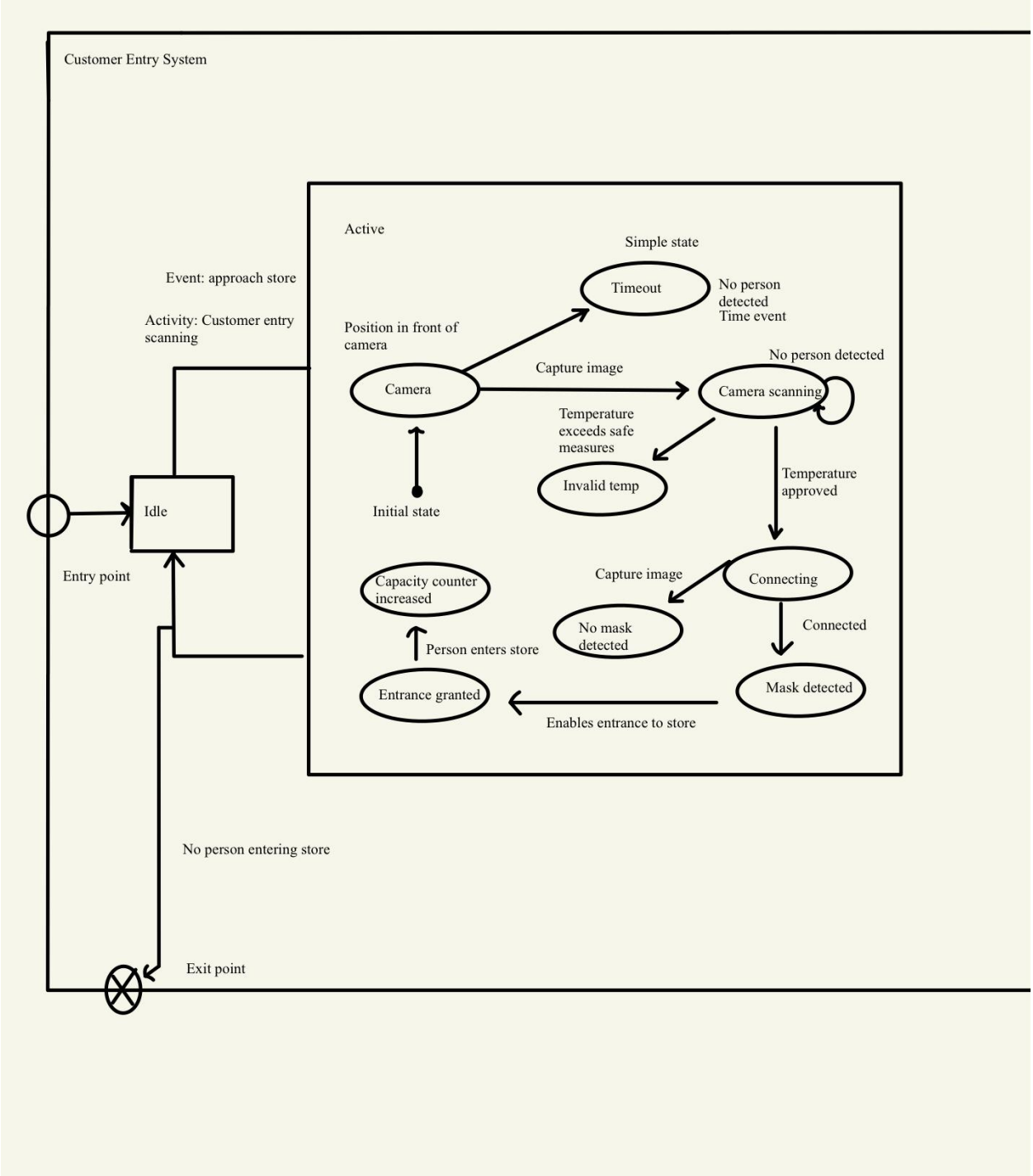


Figure 18: Entry Compliance System UML Chart

IOS application development:

The third and final component of the safeAir device system is the IOS application. The application is designed for two different audience groups: the owner and customers of a business.

By opening the application you are greeted with a login screen shown in Figure 19. You can either sign in as the owner by specifying your location and password or you have the option of signing in as a customer by just specifying the location ID you are interested in.

Figure 19: IOS application login screen.

Depending on who you sign in under the next page is slightly different. For a customer, the app allows you to see the air quality measurements for a given location by just specifying the locationID. The purpose of this is for customers to have the ability to check the air quality in a

building before deciding whether they want to enter or not. This can allow them to take measurements for their own safety to limit the chances of entering a poorly ventilated area. The measurements they have access to are included in the following figure.

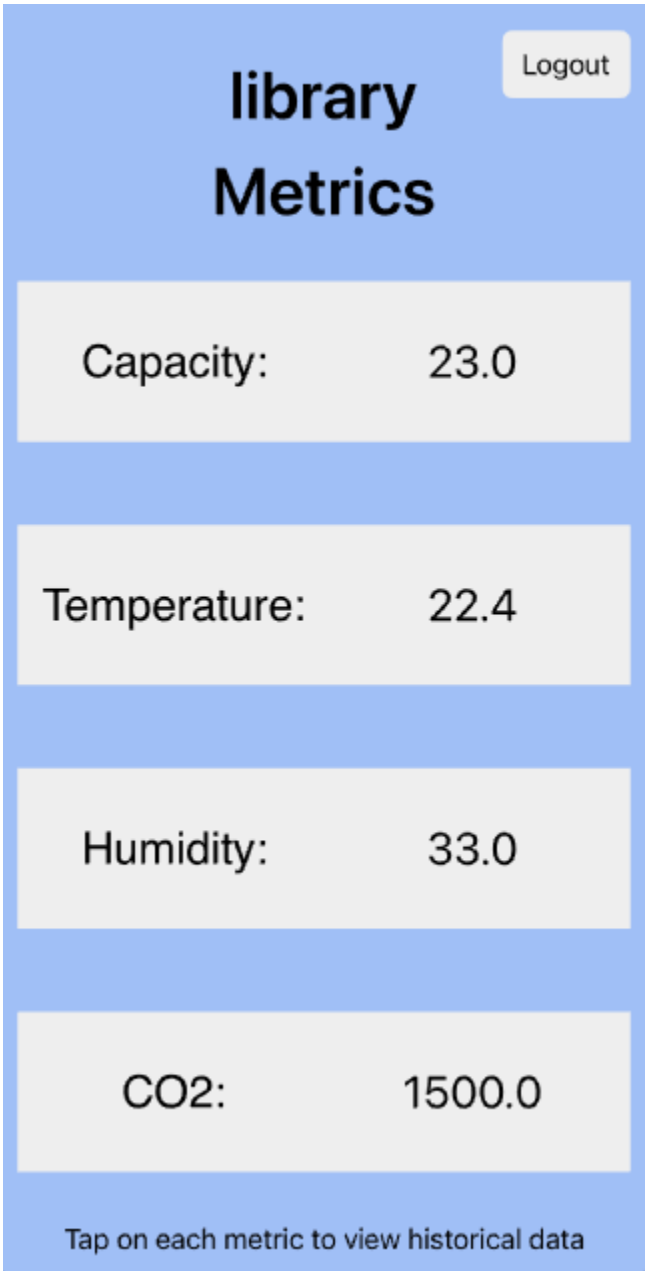


Figure 20: IOS application location metrics.

As an owner, in addition to viewing the air quality metrics you have the ability to set predefined set points that will regulate external devices if they are reached. This allows for businesses to act immediately to assure that the air quality remains at safe levels.

Experimental Outcomes:

This section will cover the evaluation and testing done over the course of the project. It will go over the requirements listed in the SRD and detail the evaluation done to verify each requirement.

1. Functional Requirements

1.1 Main Node

To evaluate the functional requirements of the Main Node, we verified that the Losant Dashboard properly uploaded changing data received from other nodes. We also verified that the node could identify a mask on a customer and notify them of their entry. These results can be seen in Figures 21 and 22.

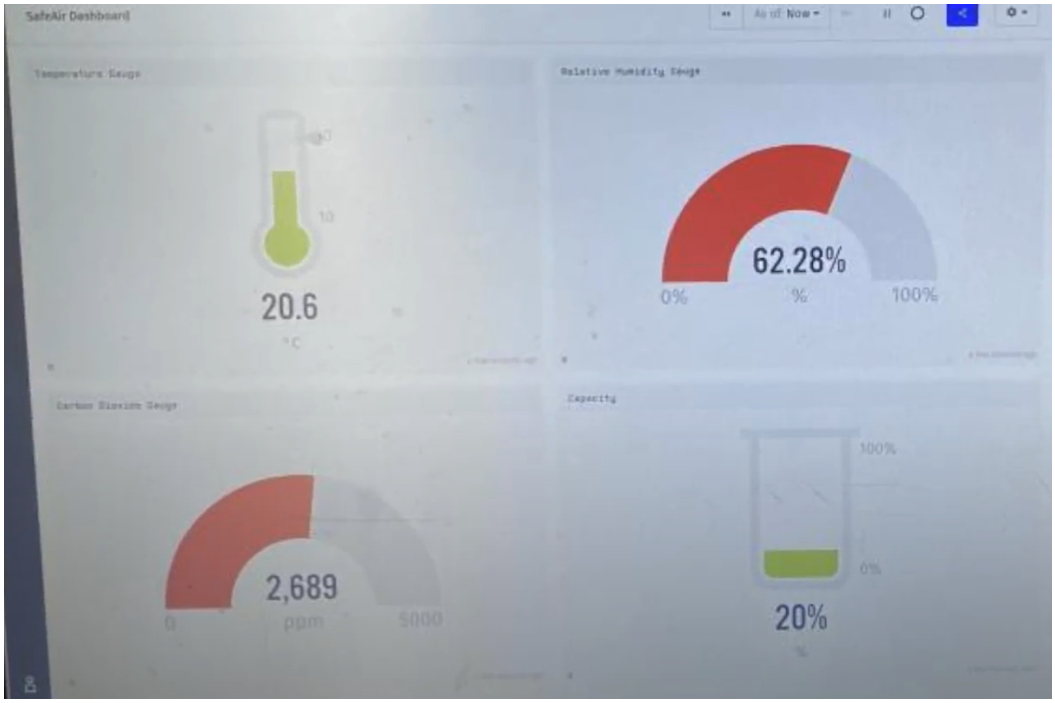


Figure 21: Losant dashboard webpage

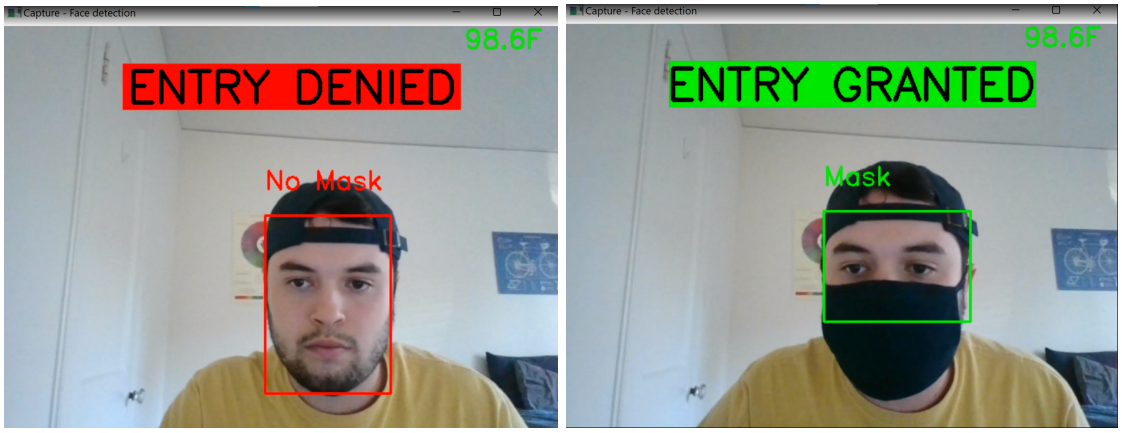


Figure 22: Screenshots of Main Node identifying customers without and with a mask, respectively

1.2 Climate Reader Node

The functional requirements of the Climate Reader Node were evaluated by verifying changes in climate reading on the Firebase database as we changed the climate. Figure 23 depicts the results of this evaluation. To begin, all three measurements remain stable. Then when we breathed onto the climate sensor, we noticed increases in all three measurements. As time passed, we see that the measurements slowly decrease and stabilize to the true room climate measurements.

Figure 23: Screenshot of the Firebase database visualizing the data measured from the climate sensor

1.3 Climate Response System

The functional requirements of the Climate Response System were evaluated by verifying that the correct regulating devices were turned on when the Climate Reader Node detects one of the measurements in a dangerous range. In our evaluation, we increased the CO2 levels in the environment by breathing onto the sensor. As a result, the Climate Reader Node alerted the Climate Response System that the CO2 levels were above the set point of safety. Thus, the Echo device activated the fan to increase airflow as seen in Figure 24.

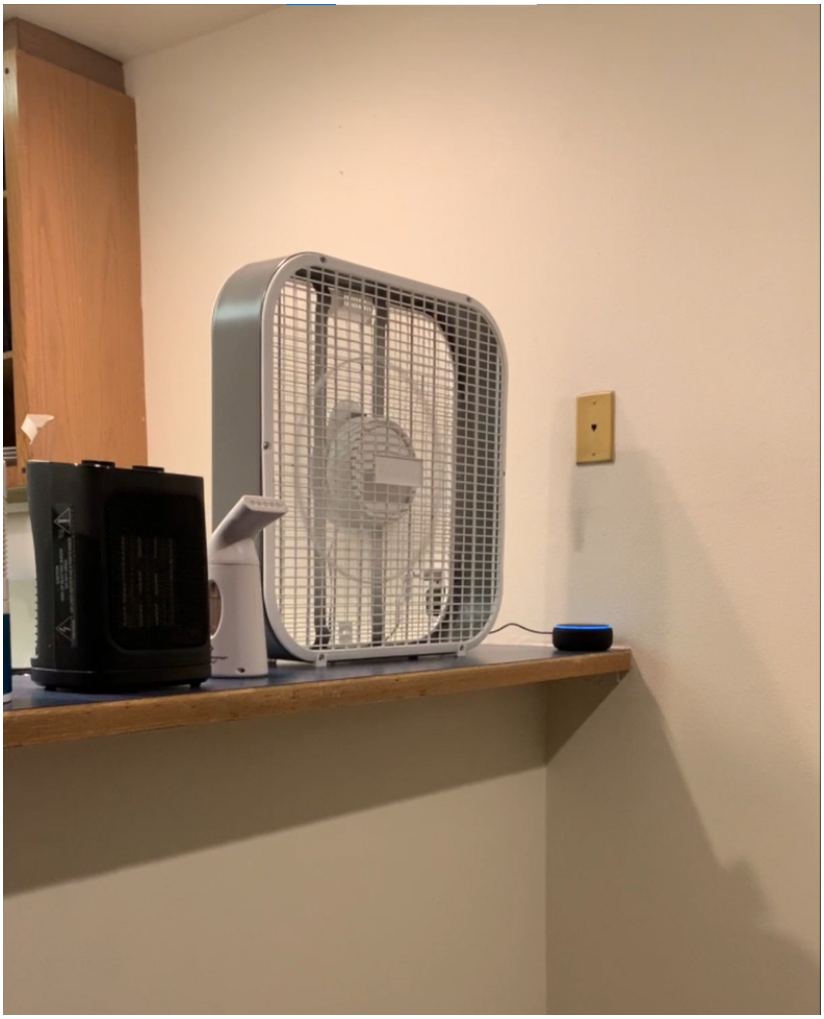


Figure 24: This picture depicts the Echo activating the fan in order to reduce CO2 levels in the environment

1.4 Capacity Tracking Node

The functional requirements of the Capacity Tracking Node were evaluated by verifying that the capacity value changed properly when we “entered” or “exited” the business. The evaluation consisted of one of our members simulating a customer entering and exiting by walking past the system in certain directions. We then observed the changes made onto our Losant Dashboard. Figure 25 verifies the change in capacity before and after walking past the system in a direction simulating a customer entering the business.

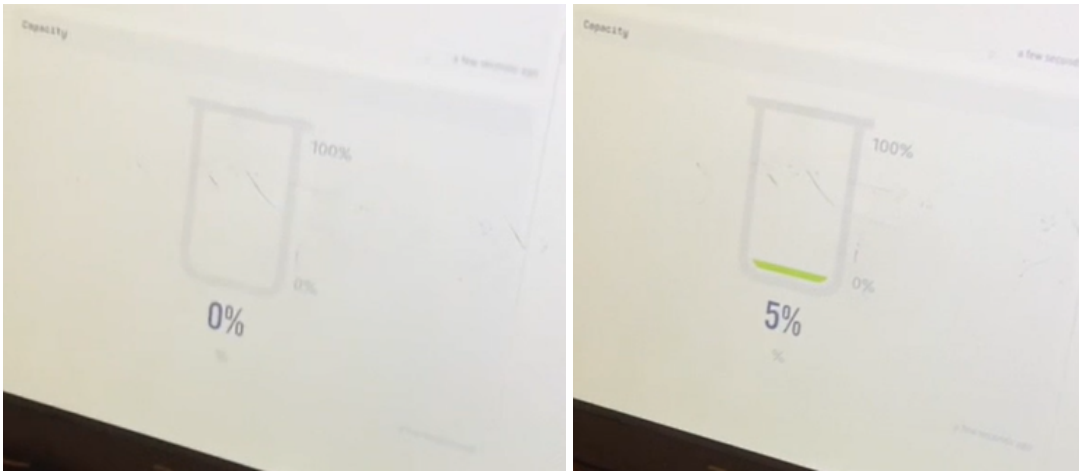


Figure 25: Pictures of the capacity measurement on the Losant Dashboard before and after simulating a customer entering. Note that the capacity limit is set as 20 so 1 customer in the business is 5% capacity.

1.5 Mobile Application

The functional requirements of the Mobile Application were evaluated by verifying that users could view current and previous climate metrics, alert owners of unsafe conditions, and determine acceptable climate conditions if they are business owners. These functions were verified by walking through our mobile application as seen in Figures 26 and 27.

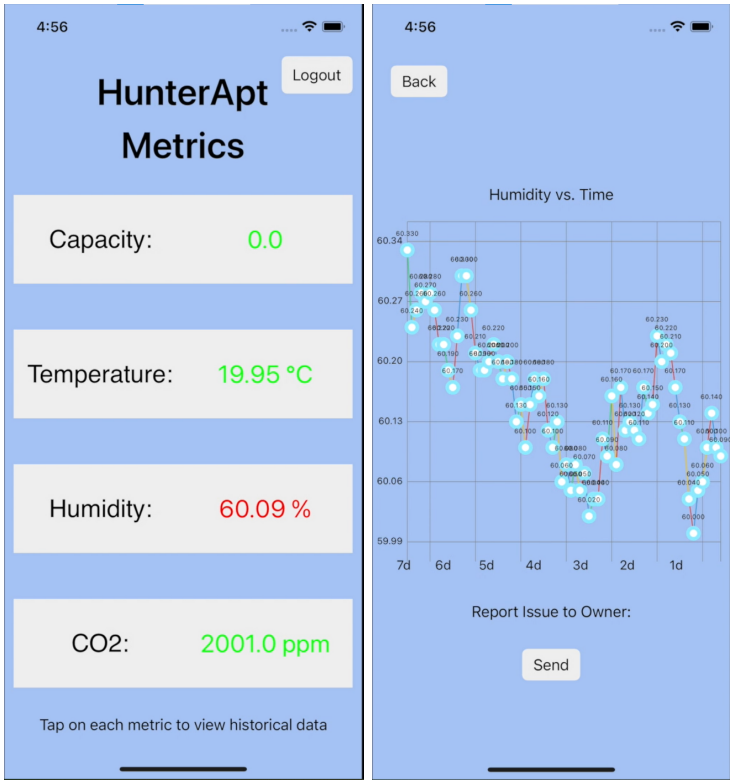


Figure 26: Screenshot of iOS app displaying the climate data and capacity

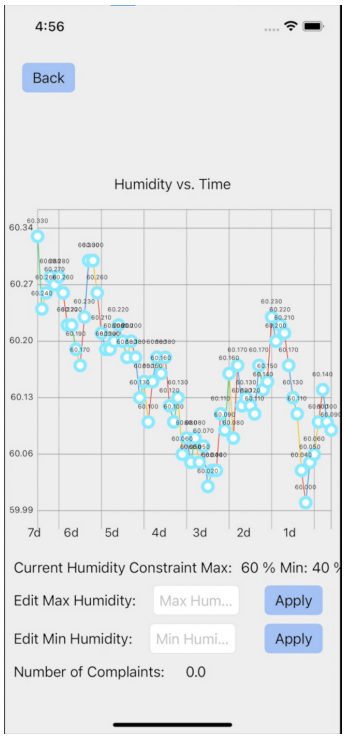


Figure 27: Screenshot showing option for business owners to set setpoints

2. Size, Weight, and Cost Requirements

2.1 Size Requirements

All nodes in our system only consisted of microcontrollers and sensors that took up less than the requirement of 1 cubic foot. This requirement excludes the main node because a monitor is needed to display the dashboard and video feed.

2.2 Weight Requirements

All nodes in our system only consisted of microcontrollers and sensors that weighed less than the requirement of 1 pound. This requirement excludes the main node because a monitor is needed to display the dashboard and video feed.

2.3 Cost Requirements

All cost specifications can be found in the Project Resources section below.

3. Mechanical Requirements

There are no mechanical requirements for this project.

4. Power Requirements

As far as specific power consumption constraints, there are no requirements. However, the monitor, main node, and each appliance will need to be plugged into an outlet to receive power. All other nodes will be powered by 5 volt batteries.

5. Thermal Requirements

There are no thermal requirements for this project.

6. Communication and Interface Requirements

6.1 I2C Communication Protocol

The I2C Communication Protocol was evaluated by following the same procedure in the Climate Reader Node. Since the climate sensor was interfaced with I2C communication, when the changes were verified on the database, the communication protocol was also verified. The verified changes can be seen in Figure 21.

6.2 Serial UART Communication Protocol and Bluetooth

The I2C Communication Protocol was evaluated by following the same procedure in the Climate Reader Node. Since the Climate Reader Node and the Main Node were connected using UART communication through bluetooth, any changes that were made in the database verifies that the protocol was working properly. The verified changes can be seen in Figure 21.

7. Control Requirements

There are no control requirements for this project.

8. Computation Requirements

8.1 Mask Tracking Algorithm

The computation requirements of the Mask Tracking Algorithm was tested by running the entire system and verifying that changes could still be viewed on the monitor in a reasonable time frame. When this evaluation was conducted, we verified that the

Raspberry Pi had the computational power to show changes in the video feed within 10 seconds.

9. Software and Firmware Requirements

9.1 Programming Languages

Peripheral nodes will use the C programming language, the main node will use Python, and the iOS application will use Swift.

9.2 Repository

All code will be maintained and accessible at the following repository: https://github.com/hnorth99/SafeAir

10. Data Storage, Format, Security Requirements

10.1 Losant Dashboard

The Losant Dashboard was evaluated by following the same procedure in evaluating the functional requirements of the Main Node. A screenshot of the dashboard correctly storing and displaying climate and capacity data can be seen in Figure 19.

10.2 Firebase Database

The Firebase Database was evaluated by verifying changes in climate data in the database. This evaluation was already done when we evaluated the functional requirements of the Mobile Application. The changes can be seen in Figure 24.

11. Precision and Accuracy Requirements

11.1 Climate Data Precision

The climate data precision can be verified on the SCD-41 datasheet.

11.2 Customer Temperature Reading

The customer temperature reading precision can be verified on the MLX90614 datasheet.

12. User Interface Requirements

12.1 Customer Entrance User Interface

The user interface requirements of the Customer Entrance was evaluated by running the entire system and verifying that customers are shown the live climate data, capacity, and video feed of the entry system. Figure 28 depicts the monitor that customers see as they enter the business.

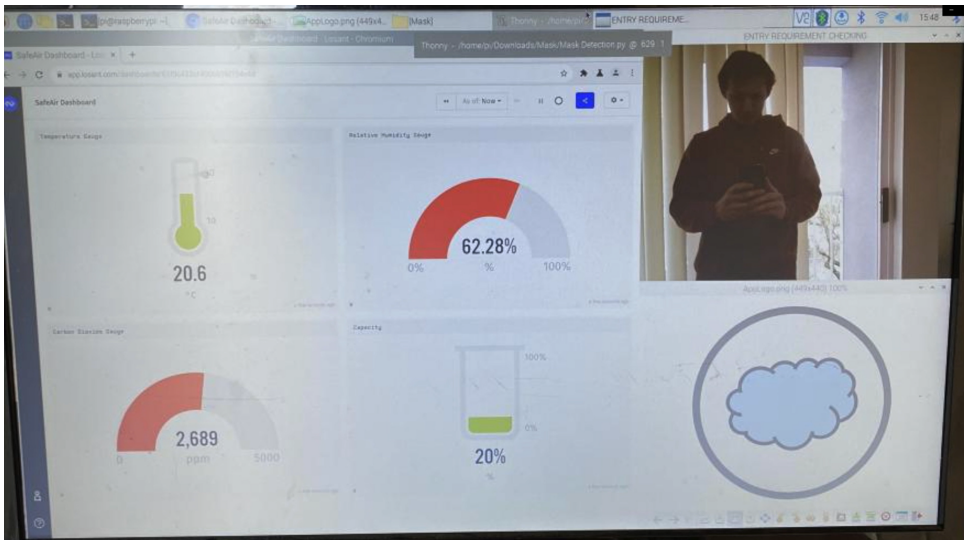
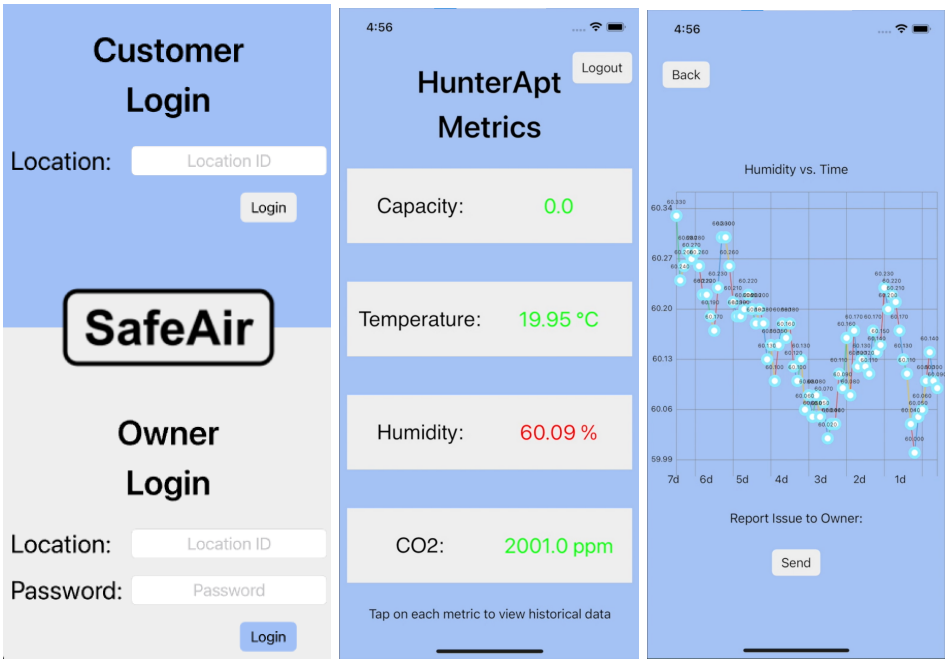


Figure 28: Picture of the monitor displaying live climate data, capacity, and video feed from the entry system

12.2 Mobile Application User Interface

The user interface requirement of the Mobile Application was evaluated by walking through the app and verifying that users can do the following: input a store or business they are planning to go to, view live metrics, and view historical data regarding the safety of the store. Figure 29 depicts screenshots taken during the walkthrough that verify that users are capable of accessing these functions.



‘Figure 29: Screenshots of the SafeAir iOS app showing that users can choose a store, view live metrics of that store, and view historical safety data of that store

Performance Standards:
It is important that this system upholds performance standards as it pertains to the health of its users and aims to lessen the spread of diseases. The temperature measuring feature must uphold medical-grade standards in order to provide accurate readings and thus accurate data regarding the safety of the facility where our device is located.

In order to be marketed, the system must be able to perform as advertised: measure capacity, temperature, air quality, and have an effective mask detection feature. The capacity must be measured as people enter and exit the room, and the air quality must be updated accurately every few minutes.

Communication and interface standards: Due to the short time span of this project, experience level of the team, and no further timeline to extend the project into the market, simple logic levels (0.0/3.3V or 0.0/5.0V) TTL logic levels are used. Higher noise margins are not necessary. Further standards are not specified due to the factors mentioned before.

Software development standards: Different components of the overall system are developed on different hardware units, by different people. To promote modularity, each distinct function of the system, is created in it's own module to be tested and therefore can be reused separately.

Impact and Consequences:
Since the onset of the COVID-19 pandemic, there has been a constant struggle between businesses and consumers. Consumers are fearful of contracting COVID-19 by entering indoor businesses. On the other hand, businesses have been hurt by the lack of willing consumers. With our product, SafeAir, we seek to develop a device that will allow consumers to feel comfortable indoors thus allowing businesses to recover financially thanks to increased consumer traffic/participation.

In an effort to alleviate consumer concern and the spread of COVID-19, businesses will hire an extra worker to stand at the entrance of their building to check that customers are vaccinated, wearing masks, and not displaying COVID-19 symptoms. Despite these checks, a-symptomatic consumers are still able to enter indoors and spread the virus. During times when the spread of the diseases is so rampant, businesses can also face government mandated (temporary) closure.

While many of these actions have been helpful in reducing the spread of COVID-19, they are detrimental from a purely economic point of view. During the initial spikes of COVID-19, the United States saw sharp increases in unemployment rates and decreases in GDP. In April of 2021, Wall Street Journal conducted a study attributing roughly 200,000 additional businesses were permanently closed as a direct result of COVID-19.
The product will seek to serve as a solution to the ongoing struggle between consumer health and economic success.

ABET Outcomes:
Outcome 4: an ability to recognize ethical and professional responsibilities in engineering situations and make informed judgments, which must consider the impact of engineering solutions in globe, economic, environmental, and societal contexts.

Recognition of Ethical Responsibility (Competent): our project expressed a competent understanding of how codes of ethics apply to the workplace. Design decisions, such as using touch-free ways of recording temperature and capacity, were made according to the problem the project was aiming to mitigate: Covid-19. Users can also interact with their phones to review measurements, and turn on appliances that help with air quality through their phones. This means users will not have to physically interact with the hardware, which mitigates not only the spread of disease, but also the potential dangers of interacting with an installed, electrical device.

Recognition of Professional Responsibility (Competent): our project expressed a competent understanding of how codes of ethics apply to the workplace. All team members' voices were valued and treated respectfully, and members did not engage in harassment or discrimination. The team avoided unlawful professional activities, disclosed conflict as they occurred, and seeked, accepted, and gave honest criticism of technical work.

Making Informed Judgements with Considerations of Impact (Competent): our project expressed a competent understanding of how codes of ethics apply to the workplace. All team members maintained and improved technical competence, undertaking tasks they qualified for or after full disclosure of their limitations and learning goals with the task. Team members also acknowledged and corrected areas of criticism after consideration from the rest of the team.

Outcome 5: an ability to function effectively on a team whose members together provide leadership, create a collaborative and inclusive environment, establish goals, plan tasks, and meet objectives.

Leadership (Competent): Team members took ownership and lead different tasks without any need for outside help. Team members were assigned their own tasks to manage and build

themselves to practice leading their own task. Members also demonstrated their tasks to one another, becoming an expert in their own area of work.

Collaboration and Inclusion (Competent): All members of the team participated fully, and each member's inputs were equally valued. Team members treated each other fairly and with respect, and did not engage in harassment or discrimination. Team members worked together to create timelines and discussed issues as they occurred. Teammates also helped each other when needed, and provided feedback on each others' work.

Setting Goals (Exemplary): Goals were fully set with additional opportunities. Clear baseline objectives were set and agreed upon during meetings based on progress made up to that point, as well as reach goals if time permitted. Goals were then further elaborated by specifying strategies and steps to achieve a task within the given time frame. Teammates kept one another accountable and supported each other to achieve goals.

Planning Tasks (Exemplary): Tasks were fully planned with contingencies. Tasks were broken down into sections with a plan on how to achieve the problem step by step. Contingencies were also discussed and agreed upon with the entire team if the baseline or reach goals could not be made. Contingencies included modifications to the requirements of a task, new materials, or time.

Meeting Milestones (Competent): All major milestones were met on time in accordance with the class timeline. All baseline demonstrations were shown as when planned at the start of the entire project. Tasks were also elaborated on in specifications that accurately portray how team members met the milestone.

Outcome 7: an ability to acquire and apply new knowledge as needed, using appropriate learning strategies.

Ability to acquire new knowledge (Competent): the team has found adequate, relevant reference materials without prompting. Team members were able to identify what knowledge they lacked but needed for the project, and could find sources independently or from each other. Team members were able to self-learn and apply new skills with unfamiliar hardware, integration, and techniques.

Ability to apply new knowledge (Competent): the team used adequate relevant external resources used in project work. Team members were able to effectively test new ideas while searching to study new solutions based on the results of a previous trial. Team members gained basic understanding of new information and performed educated strategies instead of random trial and error.

Conclusions and Recommendations:

As encapsulated in this report, the SafeAir device will be a tool for business owners to manage the safety of their patrons against COVID and other potential viruses. The SafeAir device consists of an air quality control system, a customer entry system, and a phone app to track the safety statistics of each business. The initial project proposal was designed to answer the question of how we can help businesses foster an environment against the spread of COVID-19 to keep the economy moving forward while at the same time assuring that individuals feel safe entering these businesses. From researching how viruses spread as well as COVID-19 precautions by CDC guidelines, the results of this project successfully target components of mitigating virus spread from one individual to another.

By developing the SafeAir device we as a group applied embedded systems concepts and techniques to create this real-world application that we can see businesses adopting. We learned about the Raspberry Pi and STM32F as well as other sensors and microcontrollers we included within this project through research and testing. This project allowed us to gain experience going through the complete development process from the initial design to the implementation, testing, and final design which will be beneficial for future applications and projects.

Finally, through the implementation of this project and experimental errors encountered such as low precision in sensor readings a future recommendation we would propose is to increase our funding to purchase sensors better suited for accuracy (i.e an infrared sensor that is better suited for long distances). Not only would the precision of this sensor be better, but, the reality of how an individual would take their temperature upon entering a store would be better modeled than having to get very close to the sensor. Overall, these recommendations are ways to better improve the design to limit the total amount of error as a whole.

Reference, Acknowledgement and Intellectual Property:

Our project acknowledges the following sources in developing our final product:

- MLX90614 Infrared Sensor Python library
  - https://docs.circuitpython.org/projects/mlx90614/en/latest/
- STM32F4 HAL User Manual
  - file:///C:/Users/melin/AppData/Local/Temp/dm00105879-description-of-stm32f4-hal-and-ll-drivers-stmicroelectronics.pdf
- Tutorial and Mask Detection Code: COVID-19: Face Mask Detector with OpenCV, Keras/TensorFlow, and Deep Learning
  - https://pyimagesearch.com/2020/05/04/covid-19-face-mask-detector-with-opencv-keras-tensorflow-and-deep-learning/
- OpenCV Guide to Deep Learning
  - https://learnopencv.com/deep-learning-with-opencvs-dnn-module-a-definitive-guide/
- OpenCV Library and Documentation
  - https://docs.opencv.org/
- Tensorflow API
  - https://www.tensorflow.org/guide
- Keras API
  - https://faroit.com/keras-docs/1.2.0/
- OpenCV DNN Face Detection Model
  - https://github.com/opencv/opencv_3rdparty/blob/dnn_samples_face_detector_20170830/res10_300x300_ssd_iter_140000.caffemodel
- URL Request Library
  - https://docs.python.org/3/library/urllib.request.html
- Losant MQTT Python Client
  - https://docs.losant.com/mqtt/python/
- Kaggle Mask Dataset

There is no intention to patent any part of our design in our overall project.