VEHICLE FEVER DETECTION SYSTEM

By

Rahul Rajkumar

Vinayak Dhanawade

Vishnu Rathnam

Final Report for ECE 445, Senior Design, Spring 2021

TA: Dean Biskup

05 May 5, 2021

Team 67

Abstract

The Vehicle Fever Detection System checks the temperature of passengers before entering a car. It is intended to be used in rideshare vehicles to prevent people showing signs of a fever, which is a symptom of COVID-19 or the flu, from entering the vehicle. The passenger will walk up to the device mounted in the rear back window of the vehicle. There will be one LED designating if the passenger is in proper range of the device. Red if they are out of range and green if they are in range. Once the passenger enters the correct range, the thermometer will take a reading of the passenger's temperature and a second LED will light up, green for admissible, red for not admissible. There is a second set of LEDs facing the driver on the dashboard so the driver can also see what the system is displaying. There is also a companion app that will show the exact temperature reading of the thermometer. Our system competes with other alternatives such as a digital thermometer being cheaper and a 98% accuracy when compared to a digital thermometer.

Contents

1. Introduction

1.1 The Problem

COVID-19 has disrupted multiple industries and the pandemic has shut down many companies and industries for good. One of the industries that has been negatively impacted by COVID-19 is rideshare service. These companies are most at risk due to the restriction of travel and isolation imposed by the government and social standards to slow the spread of the pandemic. Rideshare drivers are most at risk of the pandemic because they see hundreds of passengers every day. These passengers are in tight enclosed spaces with the drivers, which is the easiest way COVID-19 can spread. Not only are the drivers at risk, but also the passengers are all at risk of getting infected with COVID-19 due to ridesharing.

1.2 The Solution

Our solution is a Vehicle Fever Detection System. Our system checks the temperature of each passenger before they enter the car. This will ensure all admitted passengers do not have a fever, which is a main symptom of COVID-19. Only once our system has checked and verified that the passenger is within healthy temperature limits are they allowed in the ride share car. This helps slow the spread of COVID-19 by not allowing people with contagious symptoms to enter. Also, instead of the driver using other handheld thermometers to check each passenger's temperature, our system enables the driver to have full focus on the road as well as limiting the human contact, so they will not have to check each passenger's temperature individually. Our solution prioritizes safety and convenience for the driver and passenger.

1.3 High Level Requirements

- The thermometer must be placed in a position in which it is accessible by the passenger from a standing position with a maximum of 20° of leaning. [1]
- The temperature readings must be at least 90% accurate with a +/- .1-degree Celsius variance with the remaining 10% at a +/- .5-degree Celsius variance and provide info to the driver and the potential passenger on the display on whether the passenger is at a dangerous temperature (above 37.5° C as defined by the national institute for health research [2]) for various infrared thermometers. [1]
- Ultrasonic sensor must detect if the passenger is standing within valid distance from IR thermometer (4-6 inches) and must relay this information to the microcontroller to be displayed on LEDs to both the driver and potential passenger. [1]
- Bluetooth latency must be under 3 seconds round trip between the Bluetooth module and the phone. [1]

2 Design

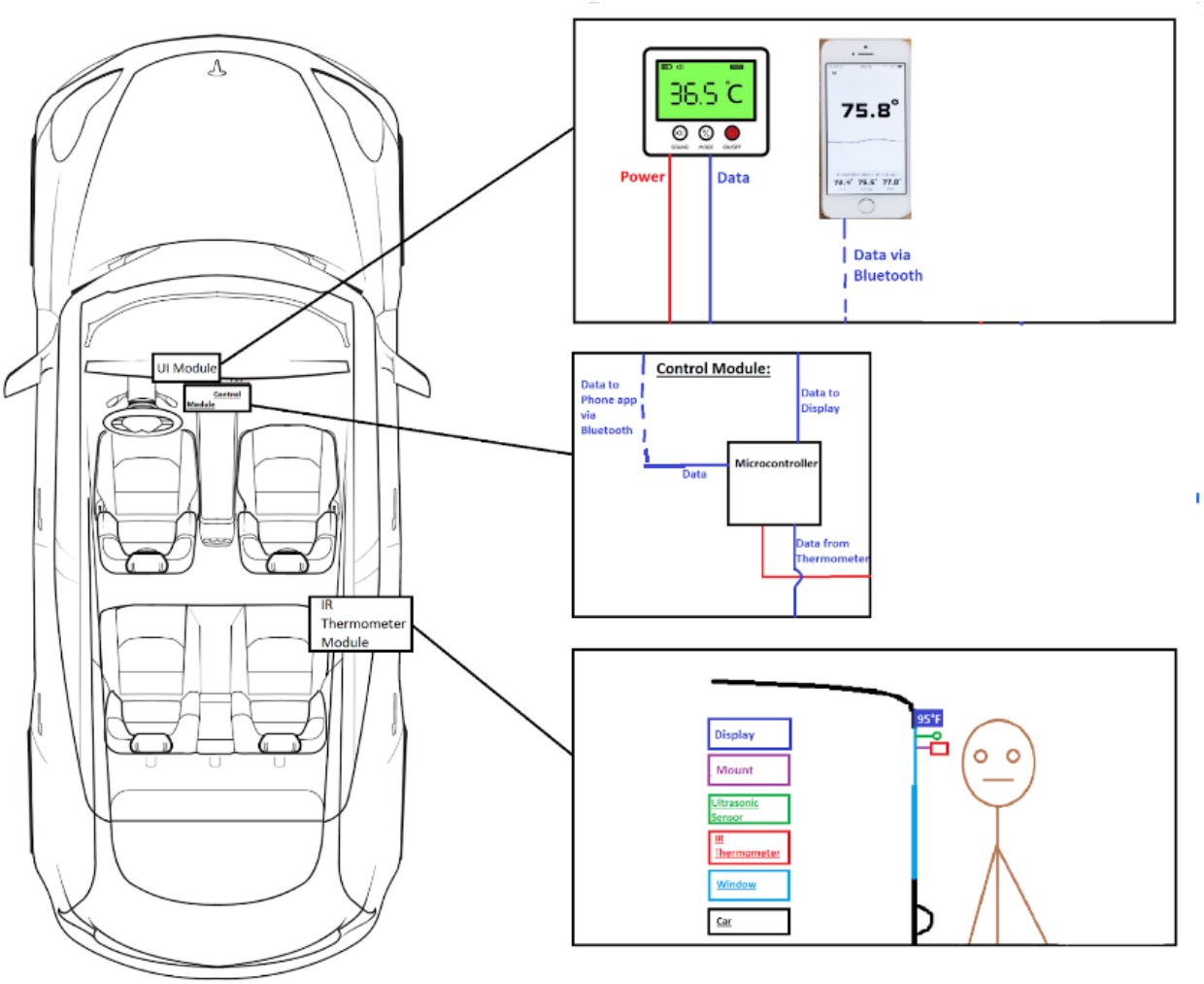

Figure 1: Graphic Model of Vehicle Fever Detection System

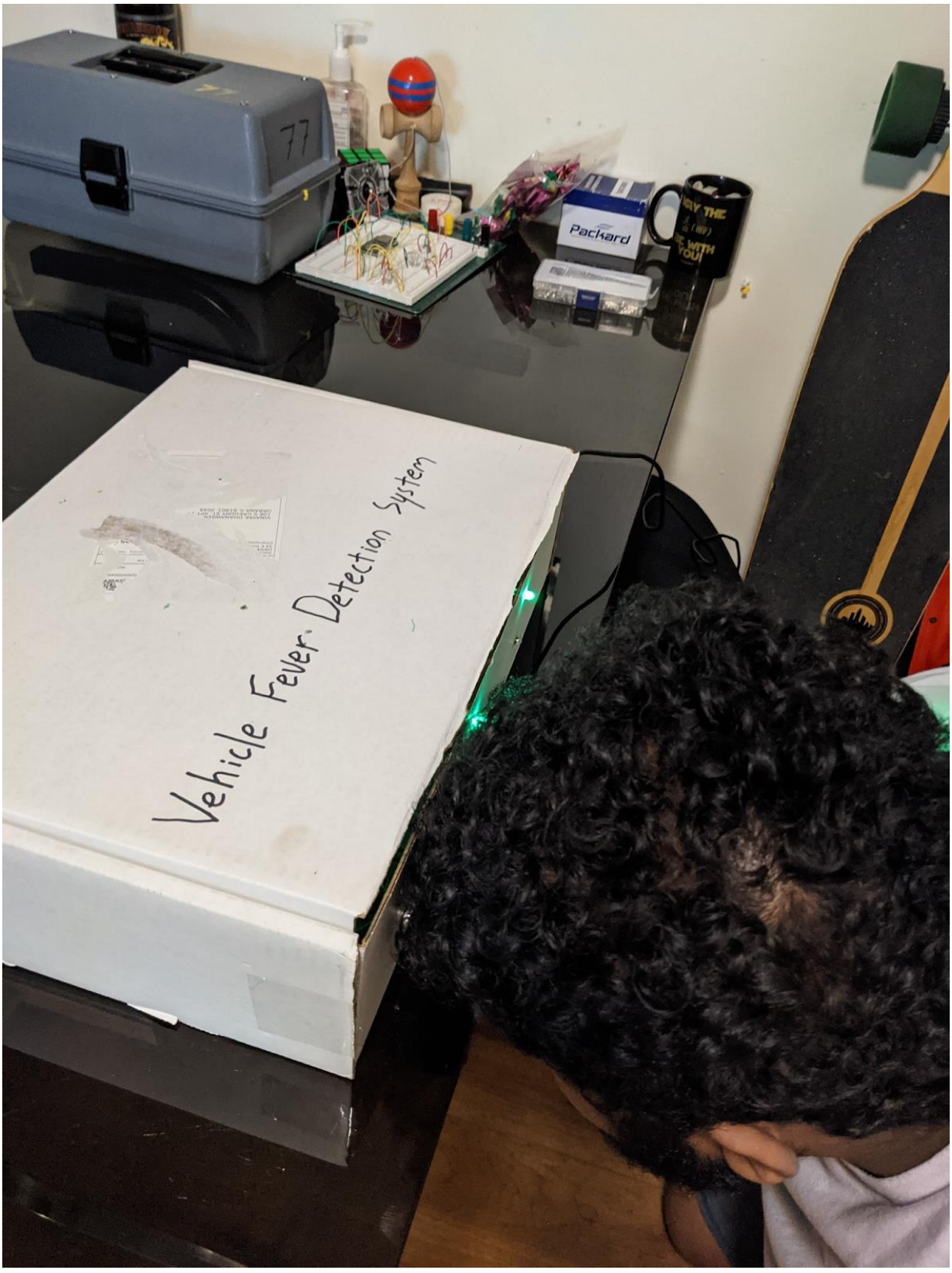

Figure 2: Final Design of Vehicle Fever Detection System

2.1 Design Procedure

The design of the system initially started with 4 modules: the UI Module, control Module, IR thermometer module, and power module. Our initial plan to design and develop our system was to develop a PCB with the functionality of a devkit, in order to gain familiarity with EAGLECAD and general PCB development. From there we planned to minimize PCB components and add breakouts to fit the specifications and functionality of our system more closely. These plans were rendered unusable, however, as we encountered shipment delays and issues in PCB design. As a result, we revised our initial design to use the ESP32 devkitc V04 instead of a PCB featuring the ESP32 microcontroller. The UI module and control module remained unchanged throughout the design process as it was not affected by any of the issues we ran into during the development phase of this project. The IR thermometer initially included two thermometers, one for the passenger, and one for the driver. However, after further consideration with our TA and team, we decided to omit the driver-side thermometer as it added no legitimate functionality to our system. The power module was entirely omitted as the devkit we ultimately used included a power module and supplied consistent voltage to our entire system. Overall, these changes in our design did not affect performance or functionality of our system, but they did incur additional cost of production.

2.2 Block Diagram

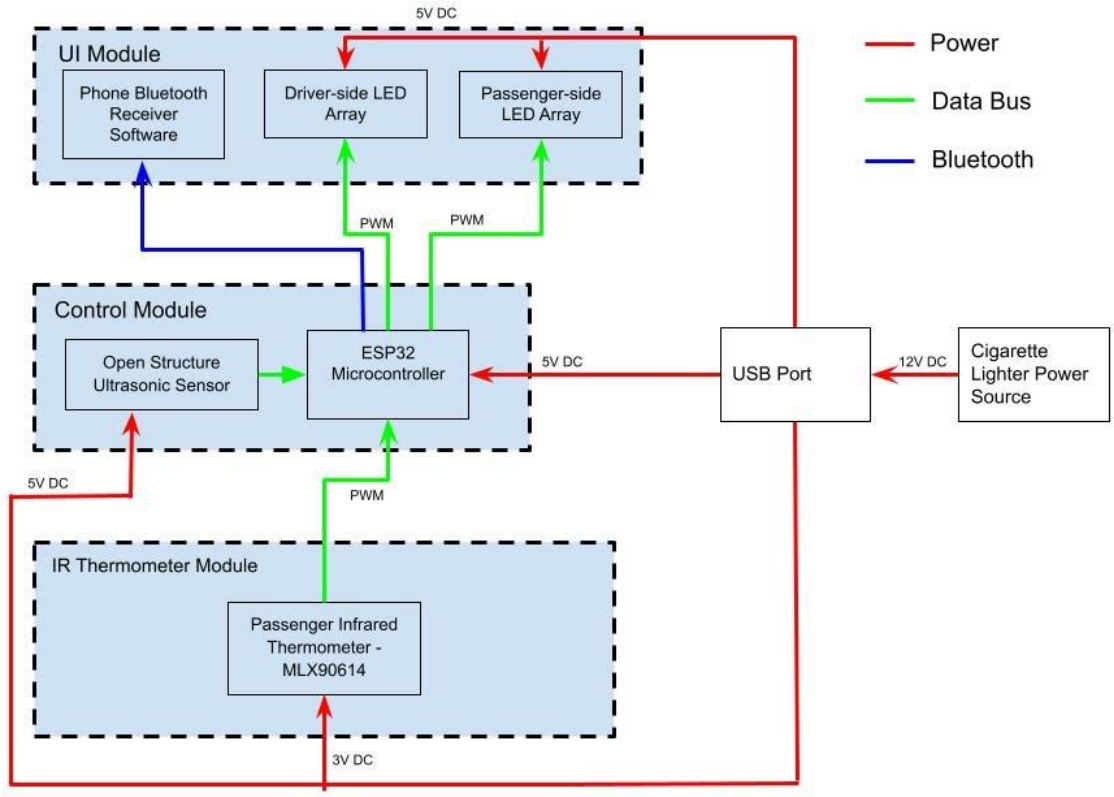

Figure 3: Block Diagram of the System

2.2.1 UI Module

This module displays the results from the thermometer effectively and cleanly. The interface is extremely user friendly and is both easy to read for the driver and the passenger. 2 sets of 2 LEDs are shown for both the driver and passenger side. One LED set shows if the passenger is in range (red for not, green for yes) and the other LED set shows if the passenger has a fever or not (red for fever detected, green for safe to go). The second LED set is only activated if the passenger is in range of the thermometer. The driver will also have a companion app on their phone that is connected to the system via Bluetooth. The driver will be able to see if the passenger is in range and what exactly the temperature reading from the thermometer is. The data will be sent via serial Bluetooth from the ESP32 microcontroller to the driver's phone on receiver software. This phone app is developed using Kotlin for android systems.[1]

2.2.2 Control Module

This module houses the ESP32 DevkitC V04 microcontroller. It processes the data from the thermometer and sends the data to the UI subsystem to be displayed. The ESP32 microcontroller accepts measurements from the HC-SR04 ultrasonic sensor to trigger temperature measurement once

the passenger is within 3 inches from the sensors. This module also makes the decision to admit or reject passengers depending on their temperature reading. It triggers the UI module actions via the Bluetooth Serial protocol in software. The system is powered through the micro-USB port connected to a USB adapter on a cigarette lighter port. Since a devkit was used instead of a PCB, a power module is included within the control module, and so our system operated on a consistent 5V supply. The control module is also responsible for the calibration of the ultrasonic sensor as well as the IR Thermometer. The control module performed the core functionality of our system via software coded on the Arduino IDE.

2.2.3 IR Thermometer Module

The thermometer module contains the MLX90614 IR Thermometer [1]that will read the potential passenger's temperature as well as the driver's temperature and send that data to the microcontroller. [1] The sensor only takes readings when the passenger is within the optimal range of about 3 inches. The sensor sends PWM signals via SDA and SCL pins to the control module, for the control module to calibrate and relay to the driver, with a recommended course of action.

3. Design Verification

3.1 UI Module

3.1.1 Bluetooth Connection

One of the requirements for the Bluetooth connection is that the round-trip communication time between the phone app and the ESP32 microcontroller must be under 3s. We were able to calculate the round-trip latency by constantly pinging the phone and measuring the response time achieved by using the devkit software. As you can tell from Figure 4, our Bluetooth connections fulfills the requirement.

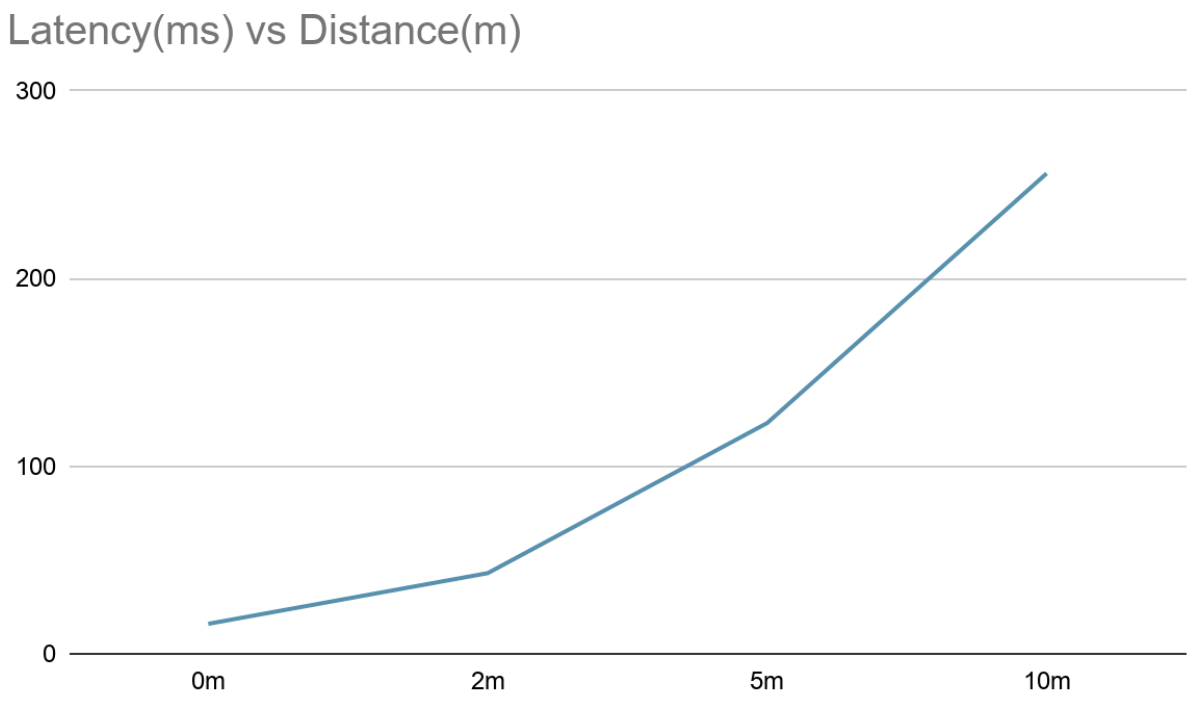

Figure 4: Round Trip Latency vs Distance of the Bluetooth Connection

3.1.2 App

Another requirement is that the smartphone application must display accurate results. Figure 5 is a screenshot is from testing the temperature of a burger. Comparing to an under-tongue thermometer, which recorded 88.1 deg F, these results are accurate within our range.

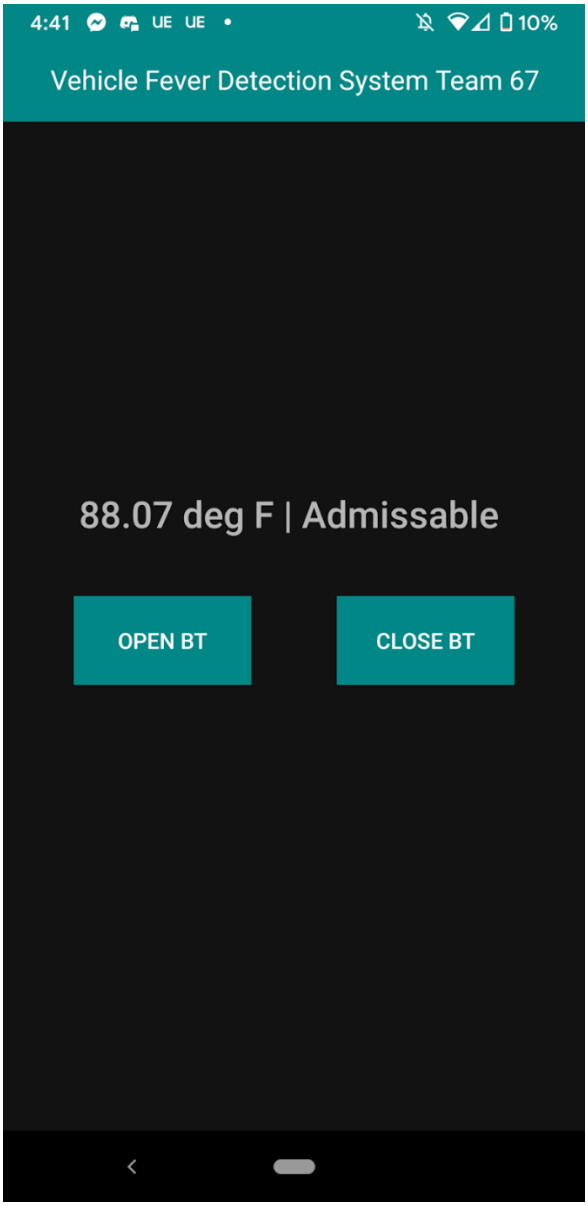

Figure 5: Screenshot of Phone App

3.2 Control Module

3.2.1 Accurate Fever Detection

Our requirement for the Control Module is that the system must classify properly if a passenger has a fever or not. In an experiment of 100 samples, 50 with a person without a fever and 50 with a hot cup of water with similar temperatures to a person with a fever, our system was able to agree with a Digital Thermometer 98% of the time in terms of properly classifying a fever or not. According to Table 1, our requirement is passed of being at least 90% accurate.

True Positive: 50	**False Positive:** 2
False Negative: 0	**True Negative:** 48

Table 1: Classification Accuracy of Properly Detecting a Fever

3.3 IR Thermometer Module

3.3.1 Accuracy of the Thermometer

Our requirement for the IR Thermometer is that it must be at least 90% accurate with a +/- .1-degree Celsius variance with the remaining 10% at a +/- .5-degree Celsius variance [1] in the optimal range. We compared the results of the IR thermometer with an under-tongue thermometer and calculated the accuracy given by the formula: $\frac{actual-measured}{actual}$. According to Figure 6, In the optimal range of 5-15cm, the IR Thermometer is upwards of 90% accurate which passes our requirement.

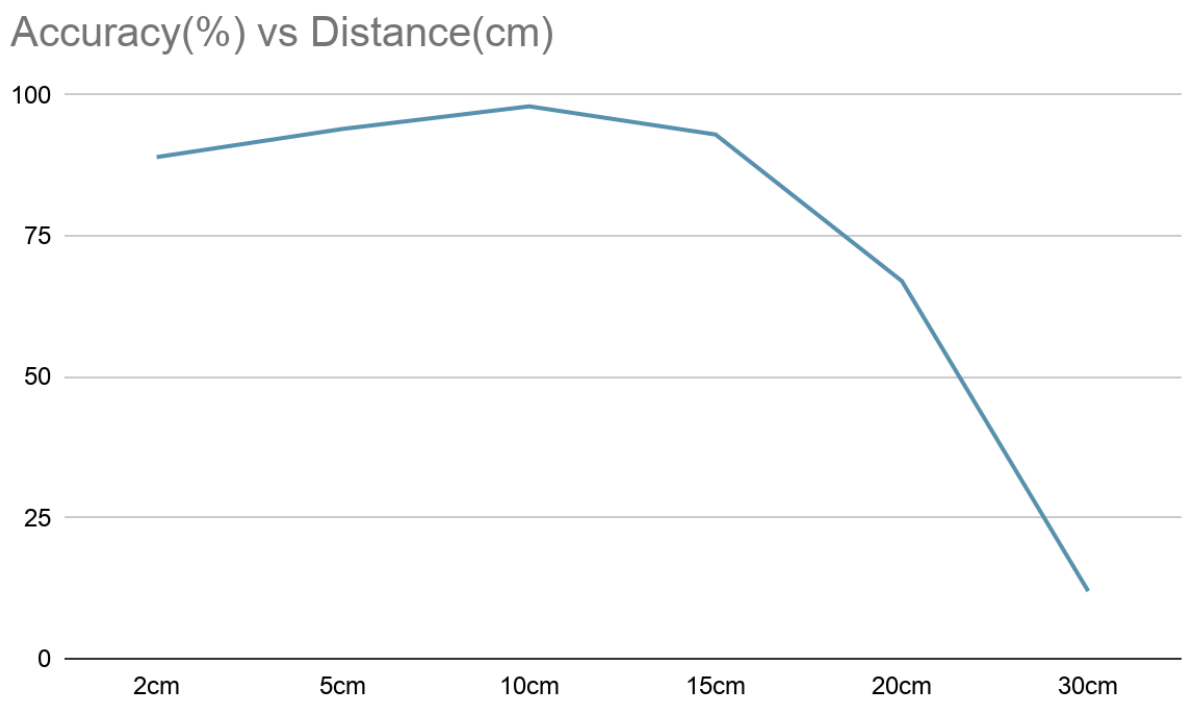

Figure 6: Accuracy of the IR Sensor Over Distance

4. Costs

4.1 Parts

Table 2 Parts Costs

Part	Manufacturer	Retail Cost ($)	Bulk Purchase Cost ($)	Actual Cost ($)
ESP32-WROOM-32 Microcontroller	Espressif	$2.99	$2.99	$2.99
875015119003 Capacitor	Wurth Elektronik	$1.68	$1.00	$1.68
0603ZC105KAT2A Capacitor	AVX	$0.20	$0.03	$0.20
VJ1206Y102JXAAC Capacitor	Vishay	$0.12	$0.059	$0.36
CRCW060312K0FKEAC Resistor	Vishay	$0.10	$0.003	$0.30
Tactile Button -SMD	sparkfun	$0.60	$0.60	$0.60
ERJ-UP8F4700V Resistor	Panasonic	$0.35	$0.035	$0.70
CRCW08055K00FKTA Resistor	Vishay	$0.15	$0.013	$0.30
2-825433-0 Connector	TE Connectivity	$2.34	$1.08	$4.68
UJ2-MIBH-G-SMT-TR Micro-USB Connector	CUI Devices	$0.45	$0.175	$0.45
CP2102-GM USB-UART Interface	Silicon Labs	$3.54	$3.54	$3.54
1N5819HWQ-7-F Diode	Diodes Inc.	$0.44	$0.06	$0.44
ESD9B5.0ST5G Diode	ON Semiconductor	$0.18	$0.02	$0.36
EEE-FN1E100R Capacitor	Panasonic	$0.33	$0.058	$1.33
C0402C104K3PACTU Capacitor	KEMET	$0.14	$0.016	$0.28
SS8050-G Transistor	Comchip tech	$0.26	$0.037	$0.52
NCP1117DT33T5G LOD Voltage Regulator	Panasonic	$0.10	$0.003	$0.50
C1210C226K3RAC7210 Capacitor	ON Semiconductor	$0.58	$0.174	$0.58
CRCW06033K30FKEAC Resistor	KEMET	$4.05	$1.23	$4.05
CRCW04022K00FKEDC Resistor	Vishay	$0.10	$0.003	$0.10
AA1608SURSK	Kingbright	$0.44	$0.095	$0.44

RED LED				
Total				**$24.4**

4.2 Labor

We calculated our labor costs at approximately $30/hr for 3 graduate engineers over the course of 16 weeks working 10 hrs/week. [1]

$$3 * 30 * 10 * 16 = \$14400$$

5. Conclusion

5.1 Accomplishments

In total, we were able to develop a working and successful system that was able to detect if a passenger had a fever or not. It also accomplishes our goal of being cheaper than the alternative digital thermometers most used today. We can accurately judge if a passenger is showing signs of a fever which is a symptom of COVID-19 and prevent them from spreading the disease, slowing the spread. Our product is very beneficial to the ridesharing industry and saves jobs.

5.2 Uncertainties

One uncertainty is mass production and selling our product to customers. We have little experience in marketing and mass production so we will need help in getting our product off the ground and in the hands of actual rideshare drivers. Another concern is the end of the pandemic. Our system will be less useful in situations where the pandemic is not a concern. Although, it is estimated the effects of the pandemic will continue for the next couple years, it is not certain that the demand of our product will be as high as it is currently.

5.3 Ethical considerations

In order to follow all the Code of Ethics as determined by the IEEE [3], we will not identify a specific individual with their temperature results and will not store any temperature data. Also, we will disclose to the patient that their temperature results will be shared with the driver but not stored or shared with anyone else. All our wiring will be enclosed in insulation and routed safely throughout the vehicle, preventing any entanglement, static hazards, or loose/hazardous electrical connections. Our unit will also be stress tested for durability and will be able to withstand air resistance travelling at high speeds without detaching from the car door and being a potential debris hazard. This aspect of our design will mostly be handled by the machine shop, or realistically, some mechanical engineering team. For the purposes of producing a proof-of-concept for our system, we will be using a simple box to house our electronics and functionality. The components that are housed outside of this box (i.e LEDs communicating information to the driver and passenger) will be housed in their own boxes, inside the car. All this combined ensures that we will be following all ethical and safety guidelines set by OSHA [4]. [1]

5.4 Future work

Our future work involves implementing our product with actual rideshare drivers. We hope to connect with companies such as Uber and Lyft to implement our product as part of a “COVID Driver Package”. We hope to continue testing with rideshare drivers and passengers to incorporate their feedback in making our system better for everyone involved. Also, we hope to mass produce and sell our product to achieve the most reach possible to end the pandemic as quickly as possible.

References

[1] ECE 445 Design Document 2021 [Online] Available: https://docs.google.com/document/d/10kKy6Q78O-QlgsWjOvPXG6XYi4sAycMS3O8uJFbk6PU/edit?usp=sharing

[2] National Institute for Health Research, 'Non-contact infrared thermometers' 2013 [Online]. Available: https://www.community.healthcare.mic.nihr.ac.uk/reports-and-resources/horizon-scanning-reports/hs-report-0025#:~:text=Two%20studies%20defined%20fever%20as,%2Dglass%20thermometer%20(17).

[3] IEEE code of ethics. (n.d.). 2021 [Online] Available: https://www.ieee.org/about/corporate/governance/p7-8.html [4] J. A. Prufrock, *Lasers and Their Applications in Surface Science and Technology,* 2nd ed. New York, NY: McGraw-Hill, 2009.

[4] Department of Labor logo United States department of labor. (n.d.).2021 [Online] Available: https://www.osha.gov/laws-regs

[5] Sparkfun, 'Infrared Thermometer - MLX90614' 2015 [Online]. Available: https://www.sparkfun.com/products/9570

[6] Instructables, 'Simple Arduino and HC-SR04 Example', 2020 [Online]. Available: https://www.instructables.com/Simple-Arduino-and-HC-SR04-Example/

[7] RandomNerdTutorials, 'ESP32/ESP8266 Thermostat Web Server – Control Output Based on Temperature', 2020 [Online]. Available: https://randomnerdtutorials.com/esp32-esp8266-thermostat-web-server/

[8] Instructables, 'ESP32: Pocket Size Distance Measuring and Logger', 2020 [Online]. Available : https://www.instructables.com/Pocket-Size-Ultrasonic-Measuring-Tool-With-ESP32/

[9] Grainger, 'Choosing and Using an Infrared Thermometer', 2020 [Online]. Available: https://www.grainger.com/know-how/equipment-information/kh-370-infrared-thermometers-qt#:~:text=Clinical%20non%2Dcontact%20IR%20thermometers%20are%20designed%20to%20be%20used,about%20two%20to%20six%20inches.

[10]Espressif, 'ESP32-DevKitC V4 Getting Started Guide', 2021 [Online]. Available: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/hw-reference/esp32/get-started-devkitc.html

[11]Espressif, 'esp32-wroom-32e_esp32-wroom-32ue_datasheet_en', 2020 [Online]. https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32e_esp32-wroom-32ue_datasheet_en.pdf

Appendix A Requirement and Verification Table

Requirement	Verification	Verification status (Y or N)
1. IR Thermometers measure temperature with at least 90% accuracy with a +/- .5-degree Fahrenheit variance with the remaining 10% at a +/- 1 degree Fahrenheit variance.	Equipment: IR thermometers, Arduino or red board, USB power supply Conduct the following steps for both thermometers: 1. a. Connect Arduino or red board to USB power supply. Use wires to connect 3.3V power and 0V from Arduino to VDD and VSS pins of IR thermometer, respectively. b. Connect SDA and SCL pin of Arduino to SDA/PWM and SCL pins of IR thermometer, respectively, using 4.7kΩ Resistor to pull up to 3.3V c. On the Arduino software, run the MLX90614_Serial_Demo to be able to observe the temperature readings from the sensor in Fahrenheit d. Gather 15 people or cuts of meat (subjects), place them one by one in front of the IR thermometer for 10 seconds each and record in a table the average measurement from the 10 seconds of data gathered for each subject. e. Perform the previous step using a traditional armpit/under-tongue thermometer instead of the IR thermometer. It is not necessary to take the average of the measurements over 10 seconds for this thermometer as the final temperature can be found in about 5 seconds. f. Compare the IR thermometer reading to the traditional thermometer reading for each test subject to ensure that the 90% accurate with a .1 degree C variance, with the remaining 10% at a +/- .5-degree Celsius variance holds throughout.	Y
1. The LEDs display the temperature results within 500ms +/- 250ms of a subject moving into position.	1. a. Using an ESP32 dev kit, a breadboard, an LED, a 220-ohm resistor, and a 4.7k ohm	Y

2. The passenger-side LEDs display the status of the passenger's positioning immediately, within 500ms +/- 250ms of changing position into less than 3 inches [5] +/- 0.5 inch range from the ultrasonic sensor. 3. Phone accurately (90% accuracy with a +/- .5-degree variance) receives temperature data via Bluetooth and displays it within 3 sec.	resistor, connect the ESP32 dev kit to a computer running Arduino IDE and wire all the components as shown in figure 4. b. As Figure 4 employs a different thermometer device than us, replace the depicted device with the MLX90614 device. To do so, connect the right and left most pin buses of the figure's thermometer to the VDD and VSS pins of the correct IR thermometer. Connect the middle bus (GPIO4) to the SDA pin of the thermometer. Connect the SLC pin of the thermometer to the CLK pin of the dev kit. c. Run code given in [6] such that when the temperature measured by the thermometer passes the threshold of 37.5° C, the LED lights up, and if the threshold is not passed, the LED stays off. d. Take 10 objects/people with temperatures above the threshold record their temperatures, and time the switching of the LED when presenting and removing these objects to/from the thermometer. e. Record these timings in a table and ensure that they stay within the desired range around 500ms 2. a. Using an ESP32 dev kit, an ultrasonic sensor, and an Arduino IDE, build the circuit depicted in figures 5 and 6 and plug the dev kit into a computer running the IDE b. Connect an LED to GPIO 2 and a resistor from the LED to power as shown in part of Figure 4 c. Upload the code found in [7] to the dev-kit but modified such that once a once the distance measured by the ultrasonic sensor within the range of 4-6 inches, the LED turns on. d. For 10 different sized objects and using a ruler, place the object just outside the 4-6 inch +/- 0.5-inch range, and move it into range whilst starting a timer. Record the time taken for the LED to be triggered on in a table and ensure the timings fall within the desired latency.	

	3. a. Set up a circuit as described in the Control Module Verification #2, but without the LED. b. Update the code used in this verification to use BluetoothSerial.h library to open up bluetooth communication and relay the IR thermometer measurements to a connected device. c. After uploading the code, open the Serial Monitor at a baud rate of 115200. Press the ESP32 Enable button to enable pairing, and pair to the dev kit from the phone. d. On the phone, open the "serial bluetooth terminal" app. This is where the data will be received and seen from the dev kit. e. Gather 10 objects or people of known varying temperatures and present them to the thermometer one by one. f. For each object, record in a table the known temperature. Once the object is presented to the thermometer, start a timer and record the time taken for the results of this measurement to show on the phone. Record the temperature received from the dev kit in the phone receiver app. Ensure that these values fall within the desired accuracy standards and latency standards.	
1. The ESP32 microcontroller must accurately determine the recommended course of action to the driver such that if the passenger's body temperature is above 37.5° C +/- .1° C, the passenger should not be allowed into the car.	1 a. Using an ESP32 dev kit, a breadboard, an LED, a 220 ohm resistor, and a 4.7k ohm resistor, connect the ESP32 dev kit to a computer running Arduino IDE and wire all of the components as shown in figure 4. b. As Figure 4 employs a different thermometer device than us, replace the depicted device with the MLX90614 device. To do so, connect the right and left most pin buses of the figure's thermometer to the VDD and VSS pins of the correct IR thermometer. Connect the middle bus (GPIO4) to the SDA pin of the thermometer. Connect the SLC pin of the thermometer to the CLK pin of the dev	Y

	kit. c. Run code given in [6] such that when the temperature measured by the thermometer passes the threshold of 37.5° C, the LED lights up, and if the threshold is not passed, the LED stays off. d. Take 15 objects/people with varying temperatures, record their temperatures, and record the behavior of the LED when presenting these objects to the thermometer, ensuring that the LED switches appropriately.	