Proceedings of the 5th World Congress on Electrical Engineering and Computer Systems and Sciences (EECSS'19)
Lisbon, Portugal – August, 2019
Paper No. CIST 120
DOI: 10.11159/cist19.120

Integration of the Raspberry Pi and Cloud Technologies for Preventing Drunk Driving

Aparicio Carranza, PhD, Elizabeth Ferreira, Jorge Martinez, Tomas Chavez
New York City College of Technology of The City University of New York (CUNY)
186 Jay Street, Brooklyn, NY, USA
acaranza@citytech.cuny.edu; elizabeth.ferreirapichar@mail.citytech.cuny.edu;
jorge.martinez1@mail.citytech.cuny.edu; tomas.chavez@mail.citytech.cuny.edu

Abstract – Annually many people die in motor vehicle accidents caused by alcohol-impaired drivers. The primary objective of this project is to develop, implement and present a way to prevent drivers from driving under the influence of alcohol. We have implemented three essential components: An electrical system using a small board computer (Raspberry Pi), connected to a MQ-2 alcohol sensor that measures levels of alcohol in the breath, similar to a breathalyzer, and an electrical foot switch that simulates the gas pedal of a vehicle which sends a signal to stop the system if the Blood Alcohol Content (BAC) reaches a threshold value over the legal limit. In addition, the data acquired by the sensor is sent to the Amazon Web Services (AWS) cloud. AWS will allow the connectivity between the entire system and the cloud through a software application Graphical User Interface (GUI), where the user uploads emergency contact information that will be used to trigger a message response to the contact via email when the BAC level in the person's breath reaches the legal limit.

Keywords: AWS, BAC, Breathalyzer, Drunk Driving, Raspberry Pi.

1. Introduction

The primary objective of this project is to develop, implement and present a way to prevent drivers from driving under the influence of alcohol. This is a major issue impacting our society in a global scale. Accidents caused by alcohol impaired drivers take the life of thirty people every day in the United States; this is one person every 48 minutes for the past 4 years [1]. These accidents not only cause pain and suffering to a large group of people, but also contribute to property damages that cost $44 billion, up to $10,000 fines and legal fees, and over a million of arrests [2]. In addition, the National Highway Traffic Safety Administration stated that about 30% of the people that die in drunk driving accidents involve a driver whose Blood Alcohol Content (BAC) was 0.08 or higher and about 68% of these accidents are caused by a driver whose BAC was 0.15 or higher [3]. It is illegal and unsafe to drive with a BAC of 0.08 or higher that is when the body experiments loss of self-control and coordination affecting balance, speech, reaction time, hearing and short-term memory loss [4]. Currently, the primary method of prevention in this matter is police law enforcement, but legal action alone cannot solve the problem. In this paper we report the outcome of a different strategy and a method of prevention that directly involves the user to act responsibly before driving under the influence by integrating the Raspberry Pi and Cloud Technology to stop people that are intoxicated from alcohol to drive. Therefore, contributing into reducing the deaths, imprisonment, and damages caused by drunk driving accidents.

To accomplish our objectives, we have made used of the MQ-2 sensor to build a breathalyzer connected to a Raspberry Pi that checks if the person's BAC is above 0.08 or below. If the BAC is above the legal limit the Raspberry Pi controls a micro-controller that signals the electric foot pedal to completely stop it from running followed by a triggered message to the person's emergency contacts which the user sets up beforehand on a web application integrated with the system powered with the Amazon Web Services. The integration of Raspberry Pi and cloud technology provides the necessary tools to accomplish a modern way to prevent drunk driving where the Raspberry Pi controls the processing unit of the device, and the cloud provides the connection and interaction between the user and the system through the web application. The implementation of this device has the potential to significantly reduce the percentage of deaths caused by drunk driving specially in the younger drivers responsible for the 27% of drunk driving crashes [5].

The rest of the paper is organized as follows: Section 1 has provided the motivation and background for the work. In Section 2, we present Components and Systems Adherence. Section 3 presents the working project, and finally our Conclusion is presented in Section 4.

2. Components and System Adherence

The integration of the Raspberry Pi and Amazon Web Services serve as the platform for this device that aims to prevent drunk driving. The Raspberry Pi is a Single-Board Computer (SBC) that runs Linux, capable of powering General Purpose Input/Output (GPIO) to control electronic components that are used to build hardware projects fundamental to the development of the Internet of Things (IoT). For this purpose, the Raspberry Pi powers the two main electrical components: the alcohol sensor and the foot pedal, and is connected to the cloud via the Internet.

2.1. MQ-2 Alcohol Sensor

The MQ-2 sensor is designed to detect Liquid petroleum gas, Propane, methane, alcohol, and other combustible steam. When the sensor perceives gas; its conductivity increases along with the gas concentration [6]. In this application of the sensor, it analyses the relationship between the ethanol in the air, so the warmer the temperature, the more reactive it is, and the reverse applies as well. Given the variation in temperature depending on the environment, the sensor is stabilized and pre-heated for a few minutes and enables the sensor to calibrate. Heating up the sensor before calibration allows for more constant temperature within the sensor itself and providing accurate data that show less fluctuation in the values in relation to the temperature around it. The module version of the sensor comes with a Digital Pin that allows it to operate without a micro-controller. To measure the alcohol, parts-per-million is used which defines how many molecules per million molecules exist in the content, connected to the Analog Pin powered with 5V to which measurements are 0V when there is no presence of gas, and elevates up to 5V when the sensor is introduced to gas. The signal output from the sensor is Analog, thus a digital conversion is needed to work with the Raspberry Pi which provides the 5V that are necessary to power up the sensor. For this, the sensor is connected to a MCP3008 chip that works as an Analog to Digital converter (ADC) which maximum power is limited to 3.3V. In a system that is running on 5V, a "step down" voltage is supplied from the signal applied by the Analog sensor to 3.3V which is controlled with a bi-directional Logic Level Converter, composed of a module with each pin corresponding to a high volt input/output as well as low voltage input/output and ground. Any of the four channels on the Logic Level converter can be used to receive the Analog signal and produce a stepped down signal. Furthermore, the High voltage on the module receives the 5V, which outputs a 3.3V on the Low Voltage; this output is wired to the pin 0 of the MCP3008 chip to convert it to a digital signal as shown in Fig. 1. After this is done, a channel is used to get the stepped down data from the Analog sensor.

The sensor's internal resistance, Rs, will change depending on the amount of alcohol within its proximity. The y-axis is the Rs/Ro ratio, which can be computed by code by first determining Ro, the resistance of the sensor in clean air. To do this, we need the Clean Air Factor. In our case, the clean air factor is approximately 9.83. These requirements are crucial since it will mathematically constraint the Ro to what it should be when in clean air, and therefore have a more acceptable and correct value. The code will determine the Rs/Ro ratio (the y-value, or coordinate). However, the BAC amount is the desired component, or the x-coordinate corresponding to the obtained y-coordinate. To get this, an equation of the line for the alcohol is determined using the first two points in the plot as presented in Fig. 2. The graph is logarithmic, which means that to get an accurate approximation of the slope, the logarithmic values are taken and converted to regular integers following equation (1). To write the equation of the line follows equation (2) resulting on equation (3) which is the equation of the line for alcohol. The slope is negative, since the graph decreases as it approaches the x-axis.

When the user blows over the sensor, like on a breathalyzer, this is the Analog signal that is applied from the sensor to the micro-controller, and it converts to a digital signal that serves as the input to the Raspberry Pi and executes the software that programs the functionality of the sensor to determine what the person's BAC is. This data is then transmitted to the micro-controller that controls the foot pedal which decides whether it is safe for the person to drive or not.

2.2. Trinket M0/ Foot Pedal

The Trinket M0 is a tiny micro-controller board that can be programmed using CircuitPython and Arduino IDE, built using the Atmel ATSAMD21 chip with a 256kB flash memory, 32kB RAM, a 48MHz 32-bit processor,

and a native USB [7]. The GPIO pins on this board include digital input/output with internally connected pull-ups or pull-downs. This board is small enough to fit into the simulated electrical foot pedal, which I/O pins can be used for Analog input and output. The purpose of the foot pedal is to stop or lock a vehicle from moving or starting up when the person is intoxicated. In other words, the electrical foot pedal is implemented to simulate a vehicle and how the communication between the alcohol sensor and vehicle interact. The board is powered up using the Raspberry Pi through USB and a Serial Communication between the Raspberry Pi and the micro-controller which receives data from the MQ-2 sensor with the person's BAC level information. Fig.3 shows a schematic of this circuitry. Once this data is received it activates the code on the foot pedal that will determine if it locks the pedal or lets the person use it depending on the presence of alcohol in the person's breath. The circuit of the foot pedal is a 3D printed pedal built with a Single Pole Double Throw Switch which ground terminal is soldered onto the GND pin of the Trinket M0, and the Normally Open (NO) terminal is soldered onto pin 0 of the board. This circuit creates a momentary switch, that allows current to flow as long a measurement of less than 0.08 BAC is received from the sensor and therefore be activated. Similarly, when the alcohol reading is greater than 0.08 BAC it will lock the pedal and not allow current to flow through, meaning that the pedal is off or inactive. On another hand, at first the Trinket M0, was a faulty, which bootloader would not start-up when it would be connected to any computer. The Red-Green-Blue (RGB) LED of the Trinket M0 would not light up. Originally the code was written in Python to work with the compiler CircuitPython, but further issues arose as it was not acting as intended where the script was not able to compile and upload to the board. A second Trinket M0 had to be implemented to successfully rebuild the circuit and upload the code. Trinket M0, can also work with the Arduino integrated development environment (IDE), after little progress was being made on the CircuitPython, the code was reworked on the C/C++ programming language to match that of the Arduino IDE. After the data is processed and the microcontroller on the pedal identified that the person is highly intoxicated with an illegal BAC limit to drive the microcontroller sends the data back to the Raspberry Pi which sends it to the Amazon Web Services cloud which is retrieved by the web application and triggers and sends a message to the person's emergency contact to alert them that this person is intoxicated and unable to drive, to take the necessary precautions to avoid accidents.

2.3. Amazon Web Services

Amazon Web Services is a secure cloud services platform, which offers a variety of services such as database storage, content delivery, etc. The main advantage of implementing cloud computing is that Amazon Web Services own and maintain the network-connected hardware required for these applications services, while the user access servers, storage and databases over the internet [8]. The web application portion of the system consists on an application built on a Serverless framework which is used to deploy web applications to the cloud provider, in this case Amazon Web Services (AWS) which is responsible for executing the code by dynamically allocating the resources. This code can be triggered using http requests, database events, file upload, etc. the code is uploaded to the cloud in the form of a function which accounts for "Functions as a Service." The services provided by AWS used in this application are: AWS Lambda, AWS Cognito User Pool, AWS DynamoDB. To build the Serverless application the use of Lambda functions is essential which runs on Node.js. The Lambda function contains information about the event that triggered this Lambda, and in the case of a HTTP request the information is specifically about the HTTP request; whereas the context object contains information about the runtime the Lambda function is taking to execute. More importantly, after the Lambda function is done, the call-back function is in charge to send the results and AWS will respond to the HTTP request with it. The way the Lambda functions are uploaded to the cloud provider is by the compression of the function and all its dependencies that are uploaded to the Storage service from AWS (S3 buckets) which is defined on a template from Serverless Framework written in YAML.

To build the backend of the application, the way the data is going to be stored is by using Amazon DynamoDB which is a NoSQL database where each table contains multiple items and each item is composed of one or more attribute. Also, another service implemented to allow the user's account to stay secure is the Amazon Cognito User Pool which allows the implementation of sing-up and sing-in functionality to the web application. So, that each user has their own personal account. Cognito User Pool supports user registration, and sign-in, identity token for signed-in users which allows the developer to control the storing of data into the database for each individual user. The front-end of the application is built using React.js which is a JavaScript library for building user interfaces that work by encapsulating components that manage their own state. Also, HTML and CSS are used for formatting and defining the backbone of the application. The application allows the user to create an account which is authenticated with the Cognito User Pool. The user will create a profile, which information is stored on the database. The type of information needed

from the user is personal information including an emergency contact that the user trusts. This emergency contact is sent to a predetermined email supported by the Amazon Simple Email Service which allows sending cloud-based email notifications. This predetermined email says that the user is trying to drive while under the influence, to contact them and take the necessary precautions to avoid them from driving under the influence.

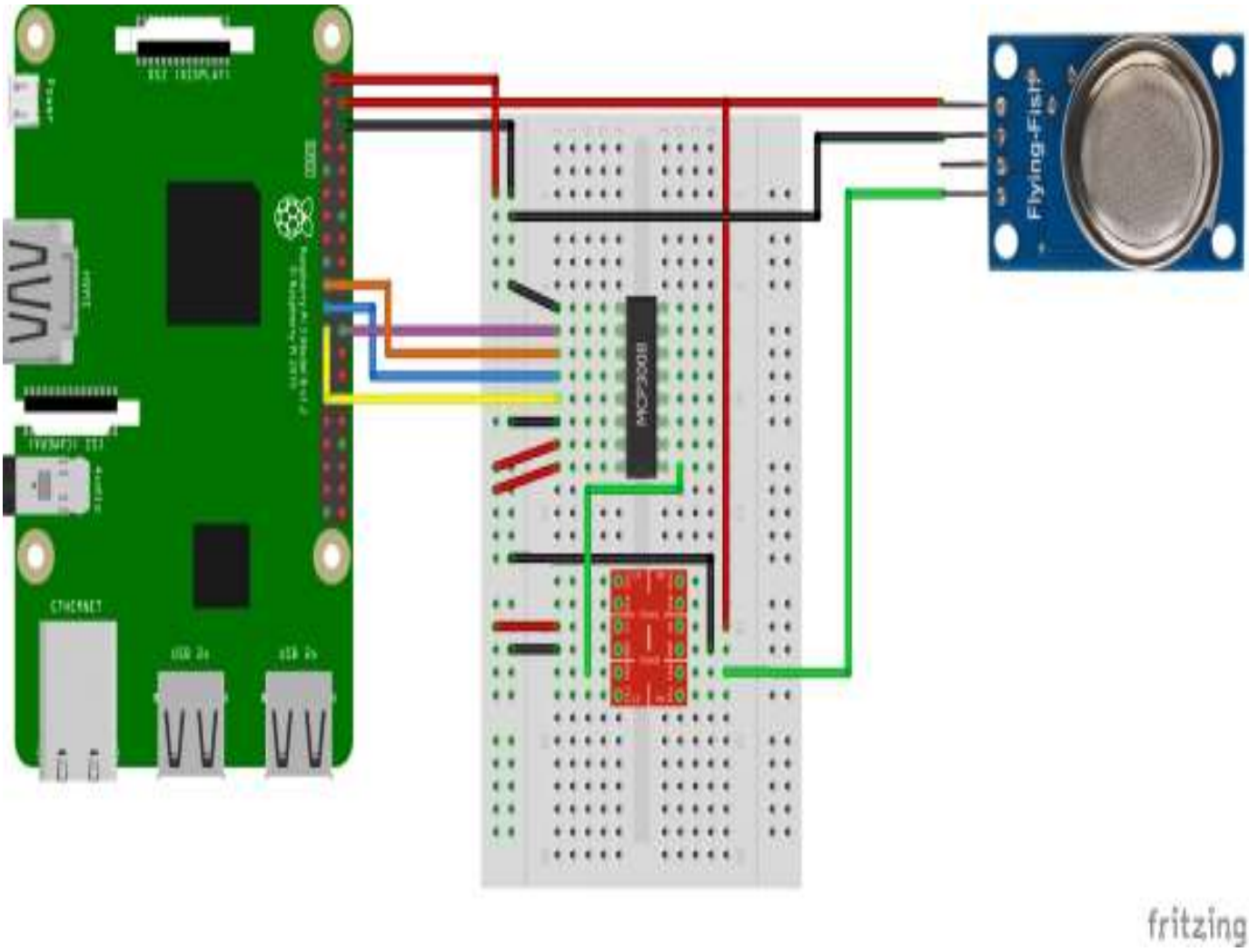


Fig. 1: MQ-2 sensor circuitry.

Fig. 1 shows the circuit built between the MQ-2 sensor and the Raspberry Pi. The sensor is directly connected to a Bi-Directional Logic Level Converter which is connected to an ADC that allows the Raspberry Pi to read the data given by the sensor. In Fig. 2 we show the Logarithmic scale plot of the gases measured by the MQ-2 sensor internal resistance (Rs) depending on the amount of alcohol within its proximity. For alcohol, it shows with a negative slope, since the graph decreases as it approached the x-axis. Where the y-axis is the Rs/Ro ratio.

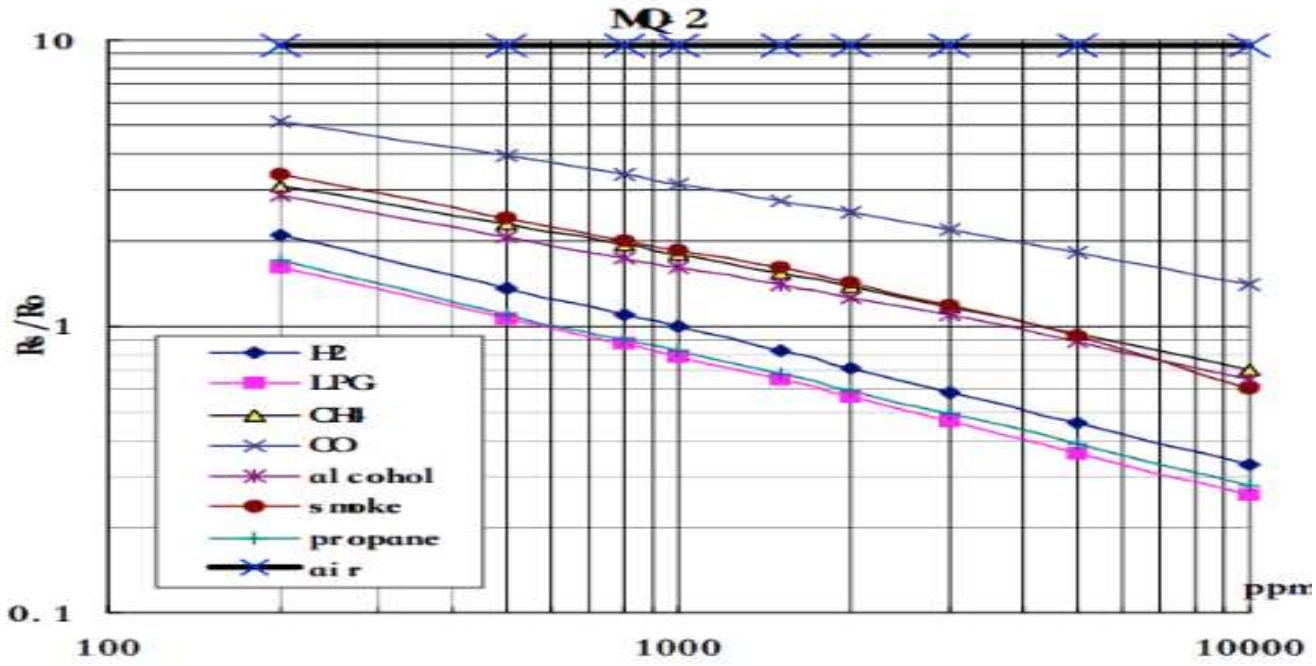


Fig. 2: Alcohol data plot.

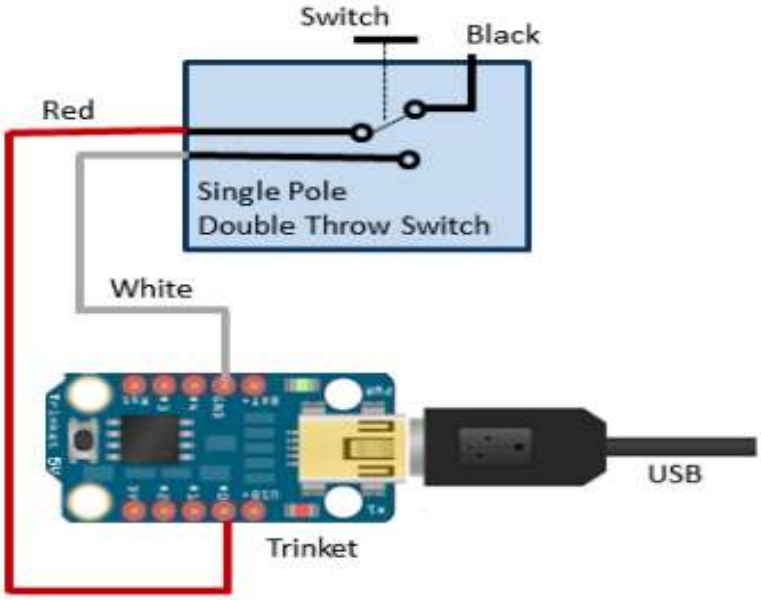


Fig. 3: Schematic of the Trinket M0 circuit.

Fig. 3 shows Trinket M0 connection to the Single Pole Double Throw Switch inside the foot pedal ratio.

2.4. Equations

The equations show how the accurate approximation of the slope of the line for alcohol is obtained given that Point1 (200, 2.93) and Point 2: (10000, 0.65) obtained from Fig. 2.

$$Slope = \frac{log(0.65) - log(2.93)}{log(10000) - log(200)} = -0.38 \quad (1)$$

$$Point[log(200), log(2.93)] \quad (2)$$

$$Y - 0.47 = -0.38(X + 2.30) \quad (3)$$

3. Experimental Results

This section focuses on a demonstration of the working project. In Fig.4 the executable code written in python is tested to measure the alcohol levels in the proximity of the MQ sensor. For testing purposes of this example, an alcohol pad with 70% Isopropyl Alcohol was used to blow into the sensor and show the results.

```
pi@raspberrypi:~/MQ_Code_4705 $ sudo python example.py
Press CTRL+C to abort.
Calibrating...
Calibration is done...

Ro=-0.119938 kohm
Alcohol Detection Level: 4.46341e-08 mg/L
```

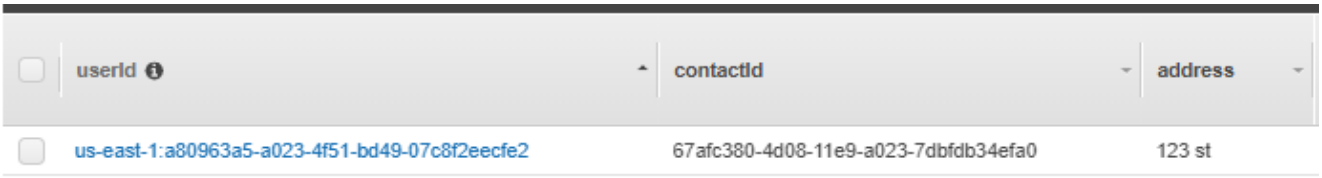
Fig. 4: Alcohol Detection code running from Raspberry Pi.

The figure shows testing alcohol detection from the Raspberry Pi terminal. First the sensor is calibrated for a few minutes and it displays the amount of alcohol present in the air close to the sensor.

Fig. 5 and Fig.6, both are demonstrations of the web application whose purpose are to offer a visual and accessible platform for the user to input their personal data and emergency contacts to prevent drunk driving by alerting the appropriate contacts to take responsible action when they are notified that a loved one is about to drive under the influence of alcohol. For a representation of the developer side, a screenshot of the database is added in Fig.7 where it shows that the information of the user is being saved in the cloud through the Amazon Web Services DynamoDB application.

Fig. 5: User Application Home Page.

Fig. 6: User Application Profile Creation.



userId	contactId	address
us-east-1:a80963a5-a023-4f51-bd49-07c8f2eecfe2	67afc380-4d08-11e9-a023-7dbfdb34efa0	123 st

Fig. 7: AWS DynamoDB table with user information.

4. Conclusion

In this paper we have shown an alternative to battle drunk driving by integrating the Raspberry Pi and cloud technology together to create a device that could prevent people from driving under the influence of alcohol. This prototype device and its individual components were successfully tested and resulted in efficiently analyzing the person's alcohol level in their breath. The software and hardware components integrated properly and functioned as expected, displaying the accurate Blood Alcohol Content level on the Raspberry Pi and the system decided whether it was safe to drive or not. The communication between the breathalyzer and the foot pedal was also successful in receiving the data and processing to determine when to deactivate the foot pedal or when it was safe to use. Furthermore, the usefulness of this device is the ability to send information to an emergency contact that provides a safety net which can ensure that the data being obtained from the breathalyzer can prevent property damages, deaths, accidents, imprisonment, and unnecessary suffering of a large group of people. Although some issues arose in the development of all components, objectives were accomplished such as simulating a car pedal to prevent the driver from starting up a vehicle, a prototype of a breathalyzer to determine the BAC of a person, and a cloud service web application that uses the data obtained from the sensor and the foot pedal to alert emergency contacts to take responsible action in the event of a person being intoxicated that tries to drive.

References

[1] National Highway Traffic Safety Administration. (2018, Nov 27). Drunk Driving [Online]. Available: https://www.nhtsa.gov/risky-driving/drunk-driving

[2] Tolber Law Centre. (2019, Feb 18). Claims Against Drunk Drivers [Online]. Available https://www.tolberlaw.com/practice-areas/claims-against-drunk-drivers/

[3] Foundation For Advancing Alcohol Responsibility (2018, Nov 22). Drunk Driving Fatality Statistics [Online]. Available: https://www.responsibility.org/alcohol-statistics/drunk-driving-statistics/drunk- driving-fatality-statistics/

[4] An American Addiction Centers Resource. (2018, Nov 20). Your Body at Different Levels of Blood Alcohol Concentration [Online]. Available: https://www.alcohol.org/effects/blood-alcohol-concentration/

[5] NHTSA's National Center for Statistics and Analysis. (2018, Nov). 2017 Data: Alcohol-Impaired Driving [Online]. Available: https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812630

[6] The IoT Hardware Enabler. (2018, Nov 27). Gas Sensor (MQ2) [Online]. Available: http://wiki.seeedstudio.com/Grove-Gas_Sensor-MQ2/

[7] Rembor, K. (2017). Creating and Editing Code Adafruit Trinket M0. [Online]. Available: https://learn.adafruit.com/adafruit-trinket-m0-circuitpython-arduino/creating-and-editing-code

[8] Amazon Web Services. (2019, Jan 2). About AWS. [Online]. Available: https://aws.amazon.com/about-aws/