CAPSTONE PROJECT

Humber College Institute of Technology & Advanced Learning

MAIDS

Meis Alarm Intrusion Detection System

Submitted by: Claudio F. Meis – N00674230

Submitted to: Mr. Kristian Medri – HC Instructor

Discipline: Computer Engineering Technology

Date Submitted: March 12, 2020

Page left intentionally blank.

Declaration of Sole Authorship

I, Claudio F. Meis, confirm that this work submitted for assessment is my own and is expressed in my own words. Any uses made within it of the works of any other author, in any form (ideas, equations, figures, texts, tables, programs), are properly acknowledged at the point of use. A list of references is included {OAECTT, 2017, Technology Report Guidelines Revised March 2017}

Signature: ______________________________

Date: January 8, 2020

Proposal

MAIDS (Meis Alert Intrusion Detection System) is a real-world Internet of Things (IoT) project aims to satisfy the following requirements:

1) Provide an affordable (under $200) home or business intrusion detection system (IDS)
2) Provide a small and easily deployed IDS for any space
3) Provide the user with a simple IDS Android-based application interface
4) Provide rapid-response and reliable multiple-channel intrusion alerts in real-time
5) Eliminate monthly monitoring fee (self-monitoring system) and no contract required
6) Provide database information for authorities use.

MAIDS is an IoT-based home/business security device built around a 3-tier model distributed computing model. It runs on a Raspberry Pi 4 Model B 1.5GHz quad-core 64-bit ARM Cortex-A72 CPU platform interfaced with a customized PCB board. The customized PCB board includes three main components:

1) A small size PCB board (34 mm x 16 mm x 15 mm) HC-SR501 Human Sensor Module Pyroelectric Infrared PIR Sensor Detector with as cone angle of less than 120º.
2) A small PCB board size (37mm x 25 mm x 20 mm) High Sensitivity Sound Microphone Digital Sensor Detection Module with an operating range between 16 to 20 kHz (1 kHz is the frequency equivalent to 1,000 cycles per second).

3) A small PCB board size (20 mm x 20 mm) SunFounder Dual Color LED Sensor Module for Arduino and Raspberry Pi capable of emitting light of two different colors (red and green); all main PCB components operate at DC 3.3V.

Remote Central access to MAIDS by a custom Android application displays an intrusion database developed with HTML, CSS, PHP and MySQL, via the Internet on a locally located Apache HTTP Server version 2.4.41 service. Furthermore, MAIDS has the capability to connect to an enterprise size wireless domain and store certificates to receive multi-channel alerts (e-mail w/ photo, SMS messages, push notification and a phone call). The prototype is surrounded by a custom 3-D and laser-cut/etched white acrylic enclosure designed for component protection.

The accompanying technical report conforms to OAECETT certification guidelines.

MAIDS' further feature development requires continued skills development in the following areas:

1) SSH Android application development with Java
2) Static IP address network configuration for Raspberry Pi 4
3) Apache server security configuration
4) Configuring Iptables or Firewalld for Linux system internal security
5) Intranet network setup and router forwarding configuration
6) KVM installation and configuration
7) Samba server setup and configuration to exchange data and files with Windows systems
8) SCP and SFTP configuration and usage.

MAIDS’ project phases (specifications, design, development and status) are reviewed by Mrs. Marilyn MacGhee, M.A., owner and operator of several Canadian-based franchise outlets in the city of Etobicoke and potentially hiring me upon graduation as her business network administrator, database developer and technological advisor. Mrs. MacGhee’s franchises have been in business for over 20 years and employ over 150 employees. Depending upon availability, Mrs. MacGhee will attend the ICT Capstone Expo.

The small MAIDS’ physical prototypes built is “…small and safe enough to be brought to class every week as well as be worked on at home. In alignment with the space below the tray in the Humber North Campus Electronics Parts kit the overall project maximum dimensions are 12 13/16" x 6" x 2 7/8" = 32.5 cm x 15.25 cm x 7.25 cm.” {Medri, 2020, Course Modules: Welcome}

In order to maintain MAIDS within safety and Z462 guidelines, it will use a maximum DC voltage of +3.3V (a Volt is the force that causes charges to move in a wire or other electrical conductor), less than 50 mA (a milliamp is equal to one-thousandth of an ampere; an ampere is the unit used to measure electric current and is a count of the number of electrons flowing through a circuit) current and maximum power consumption of 20 Watts (a Watt is a unit of power). The MAIDS’ prototype is never left unattended during transport, manipulation, displaying and modification and/or testing.

Executive Summary

Break-ins occur every 90 seconds in Canada. (King, 2019). According to research studies, homes and businesses with a monitored security system are 2.2 and 4.5 times less likely burglarized. (Woodall, 2019) MAIDS intended use as an intrusion detection alarm systems provides premise-protection (home/business) of spaces and other secure areas. Strategically placed motion/sound sensor devices within MAIDS initiates and transmits local/remote audible/visible alerts to home/business and/or police department via email with picture, Android Push Notifications, SMS Messaging and phone calls using Internet services like Twilio and PushOver. Furthermore, MAIDS' separate signal processing circuitry incorporates a sound sensing unit that detects continuous attack noises in the audio frequency range up to 20 kHz. In addition, the motion sensing unit discerns between object movement and human movement as it covers a motion cone of up to 120° and distances of up to 7 meters within operating temperature from -20° to +80° Celsius and with a low power consumption of 50 mA. The camera takes a photographic record of the intrusion event when the alarm activates attaching a caption containing time of entry, place of entry and address of home/business. Photographic/video record is then included in an email message to the owner and can help law enforcement track down potential criminals or trespassers. An Android-based phone/tablet application provides remote control capabilities. A database records intrusion information and is reachable via the Android application. Accordingly, MAIDS is capable of connecting to enterprise size networks and accept certificates. Finally, MAIDS, is enclosed in a tough acrylic enclosure designed to protect the internal sensing devices from damage.

MAIDS is an affordable, small, easily deployed and low-cost alternative that protects and secures your home or business through constant 24/7 remote monitoring using an indoor camera and motion and sound detection sensors. MAIDS provides peace of mind by providing a rapid-response and reliable multiple-channel intrusion alerts (email w/ photo, Android push notifications, phone call and SMS messaging) in real-time when you are not around. It is also capable of dispatching emergency personnel, if necessary. MAIDS also protects your pets from harm and lower Home Owner's Insurance premiums. While other competitors charge you an extravagant monthly monitoring fee between $30 and $45 dollars per month on a 42 to 60 month contract (i.e. between $1300 and $2700 per year) (Woodall, 2019) MAIDS costs less than $200! Procuring MAIDS, keeps your home and business safe at an affordable price and reduces the stress when you are not home to protect your loved ones, property and valuables.

MAIDS is a record of a successful project that demonstrates expertise in agile development processes, a strong work ethic and experience that contributes to an organization daily. It also demonstrates the software and technical skills to design and develop projects from proof of concept to deliverables by carrying out complicated tasks and duties efficiently. Moreover, MAIDS reveals an enthusiastic worker that is passionate and driven to deliver high-quality work, every time. Finally, this technical report indicates an effective communicator with strong interpersonal skills that promote cooperation and group cohesiveness, and influences business cultures affecting job performance positively which in turn aids a company's success.

Table of Contents

List of Figures

List of Tables

1.0 Introduction

A break-in is a very disturbing experience. Family, home and business are our most precious assets and their security is of great importance to all. A security system provides the peace of mind that comes from knowing that our loved ones and our possessions are protected. The consequences of a break-in can be deeply overwhelming emotionally and financially. One may recover from the financial loss but the trauma of the act perpetrated on the family and on oneself may linger for a lifetime.

Burglary is always a crime of opportunity and taking preventative measures reduces the likelihood of being affected by it. (SGI CANADA, 2020) For our families and businesses, taking a preventive measure, such as deploying MAIDS, can act as an effective deterrent against criminals from committing crimes against our family and our property. For example, when criminals see a sign informing them that security measures have been implemented, it reduces the likelihood of the crime happening. Moreover, homes and businesses with alarm systems are statistically less likely to be burglarized than homes or businesses with no security; burglars realize there is a greater chance of being caught, if activated. Finally, security alarms also reduce substantially vandalism and damage done to a person or property. An activated alarm system reduces the amount of time a thief has to commit a crime once inside the premises. In turn, the short time inside the home or business reduces harm to a person and the damage to the property and valuables, since once activated, thieves resort to fleeing the scene of the crime, promptly.

MAIDS is an affordable, reliable and simple self-monitoring home/business intruder detection alarm system based on the Raspberry Pi 4 platform, motion (Passive Infrared

Sensor) and sound sensors, and connected to the internet (IoT) to deliver home/business intrusion alerts via email (with intrusion picture), Android push notifications, phone call and SMS messaging after detecting an authenticated intrusion. MAIDS' hardware can be remotely controlled via an Android application. The software implements hardware-responsive (LED/motion/sound sensor) code that delivers human/animal intrusion detection system alerts in real-time and through a variety of communication channels.

Since most security events are initiated due to some sound that includes gunshots, aggressive behavior or breaking of glass, MAIDS includes a sound sensor. The sound sensor uses a microphone which detects the intensity of sound. The sound sensor captures the sound vibrations and changes them into signals (voltages). The output voltage then triggers the alarm system.

Moreover, a home or business intrusion involves movement on the part of the assailant, therefore, MAIDS also includes a motion sensor. A passive infrared (PIR) sensor device is used to detect a person moving in/out of the detection zone with high reliability. Positive or negative thermal radiation changes in contrast to a background are focused onto a lens that triggers the sensor element. The sensor produces an electrical output signal when the temperature of the incident radiation changes and triggers the alarm system.

Unarguably, by making your home more secure with MAIDS, you can save yourself inconvenience and money.

1.1 Scope and Requirements

MAIDS is an Internet of Things (IoT) capstone project undertaken at Humber College Institute of Technology and Advanced Learning in partial fulfillment of the requirements for the Computer Engineering Technology diploma. Its scope encompasses the design, development, modification and production of a home and/or business intruder alarm detection system.

The Meis Alarm Intrusion Detection System (MAIDS) implements a distributed computing model whose components include the following:

1. locally installed XAMPP server
2. MySQL/MariaDB database administered with phpMyAdmin, developed with HTML, CSS. PHP, and JavaScript, and accessible via the Internet
3. Capable of connecting to an enterprise wireless network and storing certificates
4. A Raspberry Pi 4 Model B 2019 embedded system with power supply and 32 GB SDHC card fitted with a custom PCB board containing a motion sensor, a sound sensor and a dual LED module
5. A custom acrylic, 3-D and laser cut/etched enclosure fitted with a small (3 mm x 3 mm) fan for cooling and an ON/OFF switch.

Furthermore, the project is documented by following and producing an acceptable OACETT technical report. Also, it is important to emphasize that the MAIDS project was not CSA tested.

The specifications for the MAIDS software-hardware components are as follows:

1. Raspberry Pi 4 Model B 2019:

a. CPU: 1.5GHz quad-core 64-bit ARM Cortex-A72

b. RAM: 4GB of LPDDR4 SDRAM

c. Ethernet: Full-throughput Gigabit

d. Wireless: Dual-band 802.11ac wireless networking

e. Bluetooth: 5.0

f. USB ports: Two USB 3.0 and two USB 2.0 ports

g. GPU: Dual monitor support, at resolutions up to 4KVideoCore VI graphics supporting OpenGL ES 3.x4Kp60 hardware decode of HEVC video.

2. HC-SR501 Human Sensor Module Pyroelectric Infrared PIR Motion Sensor Detector:

a. Product Type: HC--SR501 Body Sensor Module

b. Operating voltage range: DC 4.5-20V

c. Quiescent Current: <50uA

d. Trigger: L (Default repeated trigger)

e. Delay time: 5-200S (adjustable)

f. Block time: 2.5S (default)

g. Angle Sensor: less than 120º cone angle

h. Lens size sensor Diameter: 23mm (Default)

i. PCB board size: 3.7 cm x 2.5 cm x 2cm (1.46x0.98x0.79inch)

j. Digital output pulse high (3.3V DC) when triggered (motion detected) and digital low (0V) when idle (no motion detected)

k. Sensitivity range between 7-20 feet (3-6 meters).

3. Sound sensor:

a. Operating range between 3.3 V DC to 5.0 V DC

b. Operating current between 4-5 mA

c. Voltage gain of about 26 dB

d. Impedance of 2.2k Ohms

e. Frequency range between 16 to 20 kHz

f. Noise to signal ratio of 54 dB

g. Output model: digital switch outputs (0 (low) and 1(high))

h. PCB size: 3.4 cm x 1.6 cm.

4. Dual Color LED Module:

   a. Dual-color LED: red and green

   b. Common cathode

   c. PCB size: 2.0 cm x 2.0 cm

   d. 3-Pin anti-reverse cable

   e. Working voltage: 3-5V DC.

5. Rocker Switch - SPST (round):

   a. Rated up to 10 A at 125 VAC

   b. Two-Pin switch

   c. Size: 20 mm diameter.

6. Enclosure:

   a. *Design:*

      i. CorelDraw 2018 landscape .pdf file

      ii. Stroke width: hairline (0.000 mm)

      iii. Outside laser cut: green

        iv. Inside laser cut: red

        v. Etching color: black with stroke width thicker than hairline.

    b. *Physical Characteristics:*

        i. Small footprint (100 mm x 67 mm x 100 mm)

        ii. Hollow shell WT2447-1-8/1212 Acrylic White Sheet

        iii. Translucent: 10%

        iv. Thickness: 3 mm (1/8")

        v. Glossy/shiny surface

        vi. Weatherproof/UV stable

        vii. Etched icons and lettering

        viii. 3-D printed camera holder bracket

        ix. Laser cutouts: Round rocker switch, SDHC card, fan (30 mm x 30 mm), USB2.0, USB3.0, Ethernet, 2 micro HDMI connectors and audio ports.

7. Raspberry Pi 4 Compatible Power Supply:

    a. ON/Off Switch

    b. 5V 3A USB-C Charger Adapter for Raspberry Pi 4 Model B 2GB Version.

8. Storage:

    a. Sandisk Extreme Pro: 32GB SDHC UHS-I Card (SDSDXXG-032G-GN4IN)

9. Android Device:

    a. Display: 9.60 inch (800 x 1280 resolution)

    b. Processor: 1.3 GHz quad-core or higher

    c. Front Camera: 2 MP

d. RAM: 1.5 GB

e. OS: Android

f. Storage: minimum 8 GB

g. Rear Camera: 5 MP

h. Battery Capacity: 5000 mAh.

10. MySQL/MariaDB Database:

   a. *Hardware:*

      i. CPU: Intel Core or Xeon 3GHz (or Dual Core 2GHz) or equal AMD CPU

      ii. Cores: Single (Dual/Quad Core is recommended)

      iii. RAM: 4 GB (6 GB recommended)

      iv. Graphic Accelerators: nVidia or ATI with support of OpenGL 1.5 or higher

      v. Display Resolution: 1280 × 1024 is recommended, 1024×768 is minimum.

   b. *Software:*

      i. Windows 7 (64-bit, Professional level or higher)

      ii. Mac OS X 10.6.1+

      iii. Ubuntu 9.10 (64bit)

      iv. Ubuntu 8.04 (32bit/64bit)

      v. Fedora 11 (i386/x64)

      vi. Microsoft .NET 3.5 Framework

      vii. Cairo 1.6.0 or later

viii. glib-2.10, libxml-2.6, libsigc++ 2.0, pcre, libzip.

11. XAMPP Stack 7.4.1-1:

i. PHP 7.1.1

ii. Apache2.4.25

iii. OpenSSL 1.0.2j

iv. MariaDB 10.1.21

v. Perl 5.16.3

vi. OpenSSL 1.1.1d (UNIX only)

vii. phpMyAdmin 5.0.1.

12. Programming Languages:

a. HTML 5.0

b. CSS 3.0

c. JavaScript ECMAScript 2019.

2.0 Background

Break-ins occur every 90 seconds in Canada and more than 80 percent of break-ins occur during daylight hours. (SGI Canada, 2020) Statistics from 2018, showed that there were 159,812 burglaries across Canada and all types of properties. That is 431.24 reported burglaries per 100,000 persons. In other words, 4 percent of all Canadian households were burglarized; that is to say 1 out of 28 households were burglarized across the country. (Statistics Canada, 2018) Furthermore, once burglarized, a home or business is 12 times more likely to be burglarized again! (Woodall, 2019, Canadian Crime Rates Burglary & Home Invasion: A Real Threat)

Closer to home, in the City of Toronto, Statistics Canada reported that in 2018 there were 14,265 Break and entering cases reports; that is 227.36 reported burglaries per 100,000 persons, an increase of 3.81 percent from 2017. According to the report, 1,723 persons were charged with the offence; 145 youths between the ages of 12 and 17 years of age were charged. (Woodall, 2019, Canadian Crime Rates Burglary & Home Invasion: A Real Threat)

More disturbing, according to the Canadian Centre for Justice Statistics, a typical home/business invasion robbery in Canada is carried out by strangers 68 percent of the time in which a weapon is present 62 percent of the time (firearms 33 percent, Knives or cutting instruments 30 percent other weapons 42 percent) during the home/business invasion and victims sustain injuries in 50 percent of the cases. (Statistics Canada, 2002)

Taking into account the abovementioned statistics, it is not surprising that home/business owners are looking for a home intrusion detection system to safeguard their families and property. However, they are finding it more difficult to protect their families, homes and businesses with the skyrocketing pricing of commercial alarm systems. For example, Vivint will charge around $700 just for the basic starter equipment package and a monthly monitoring fee between $30 and $45 dollars per month on a 42 to 60 month contract; that is between $1300 and $2700 per year! {Safety.com, 2020)

According to research studies, homes with a monitored security system are 2.2 times less likely to be burglarized and business with a monitored security system are 4.5 times less likely to be burglarized. (Canadian Centre for Justice Statistics, 2019) In addition, 85 percent of police chiefs recommend the installation of monitored security systems. Furthermore, in its criminology study, the University of North Carolina at Chapel Hill found that thieves check for alarms in a home or business 83 percent of the time and if one is found more than 50 percent of them are deterred from committing the break-in. (Canadian Living, 2019)

3.0 Methodology

3.1 Required Resources

The MAIDS project was guided by the following resource management principles:

1. The use of fewer components, parts and materials, as possible
2. To use tools, methods and materials to reduce material and energy consumption during the lifecycle of the project and, in particular, to remove, as much as possible, hazardous substances from the production process.

3.2 Parts, Components, Materials

3.2.1 Parts

Parts are the distinct pieces comprising the MAIDS project which are manufactured separately and used to build or repair said project, and when combined with other pieces makes up the whole. This section provides information about the parts that make up the MAIDS assembly.

1. It includes multicolored Dupont Wire Female to Female Breadboard Jumper Wires Ribbon Cables Kit for Arduino which serve to internally connect all the various MAIDS components.
2. A 5 volts, 3 ampere power supply version compatible with the Raspberry Pi 4 Model B 2 GB (gigabyte) module with an ON/Off Switch and a USB-C Charger Adapter to provide the power needed for the Pi module.
3. A brushless cup Raspberry Pi 4 Cooling Fan with dimensions of 30 mm x 30 mm x 7 mm, with output voltage of 5 volts DC (direct current) and four

rectangular aluminum heatsinks used to dissipate heat from running component (thermal control) and maintain moderate operating temperatures.

4. A SanDisk Ultra 32 GB, micro Secure Digital High Capacity (SDHC) flash memory card based on the SDA 2.00 specification, Ultra-High-speed (UHS-I) Card running at approximately 104 Mb/s with Adapter in order to hold Raspbian operating system (OS) and serve as an storage device for produced data.
5. A Panel Mount, round, Snap-In, single pole single throw (SPST) (a switch that has two inputs and two outputs; each input has one corresponding output) rocker switch with maximum alternating current (AC) of 16 amperes and maximum operating voltage of 125 volts used to turn MAIDS on or off.
6. A white acrylic, laser-cut joints used to hold the enclosure together
7. A custom 3-D printed, resin-based camera holder bracket used to attach and hold the MAIDS USB camera in place.
8. A set of four M2-M3 thread, Male-Female connectors, nylon, hexagonal shapes, standoffs with plastic thread that serve to attach and separate modules, isolate circuit components and prevent short circuits.

3.2.2 Components

The MAIDS alarm system is comprised of a few components, most modular in nature. Modular components are combined to work together and form a single functioning unit. Since each module is separate, it is often possible to upgrade, change or repair one component while leaving the main system operational. The main components of MAIDS include the following:

1. A white acrylic base and laser-cut and etched enclosure used to protect the internal components and designed specifically to provide good aesthetics.
2. A high sensitivity, high reliability, low voltage and low power consumption HC-SR501 Human Sensor Module comprised of a pyroelectric infrared (PIR) Sensor used to detect changing levels in infrared radiation, like the radiation changes of a moving (human) body.
3. A high sensitivity Microphone Audio Amplifier Module with an operating output of 20 decibels (dB) gain, low noise and running at 3.3 volts to 5 volts DC and used to detect sound and changing sound intensities.
4. A SunFounder, dual color light emitting diode (LED) sensor module for Arduino and Raspberry Pi used to provides visual intrusion information; green=safe and red=intrusion).
5. A Raspberry Pi 4 Model B 2019 module, equipped with a Quad-Core, 64 Bit central processing unit (CPU), with WiFi and Bluetooth capabilities and four GB of RAM. The Raspberry Pi module is used to process and control all sensor I/O signals.

3.2.3 Materials

Proper materials selection insures that the right material is used for the right job. Usually, materials are divided into four key groups: metals, polymers, ceramics, and composites. To complete the MAIDS project, metal and polymer components were the main materials used as inputs to the manufacturing process.

1. The project made use of MG Chemicals, 5" x 3" Copper Clad Board, Double Sided (both the top surface and the bottom surface are coated with a conductive material), one ounce Copper with a board thickness of 1/16" and FR4 designation (NEMA grade designation for glass-reinforced epoxy laminate material) used in the fabrication of the custom-made project circuit PCB board.
2. Lead Free Solder Wire with a Rosin Core of 0.6 mm in diameter used to provide safe soldering conditions of parts and components.
3. A Falken Design WT2447-1-8/1212 Acrylic White Sheet with a translucence of 55 percent, board size of 12" x 12", thickness of 1/8" and used as the main material for the enclosure, joints and base plates.

3.2.4 Bill of Materials (BOM)

TABLE 1 BOM FOR MAIDS PROJECT.

	Designator	MPN	Qty	Link
1	HC-SR501 Human Sensor Module Pyroelectric Infrared PIR Sensor Detector PIR Sensor	4260474030781	1	https://www.amazon.ca/Motion-Detector-Sensor-Switch-Arduino/dp/B075CNDXTB
2	High Sensitivity Microphone Audio Amplifier Module Output 20dB Gain Low Noise DC 3.3V/5V	STK0114016717	1	https://www.amazon.ca/Microphone-Controller-Detection-Sensor-Arduino/dp/B01DBGZ2K2
3	SunFounder Dual Color LED Sensor Module for Arduino and Raspberry Pi	2SSR	1	https://www.amazon.ca/SunFounder-Sensor-Module-Arduino-Raspberry/dp/B014KR6VBA
4	MG Chemicals 5" x 3" Copper Clad Board, Double Sided, 1 oz Copper, 1/16" Thick, FR4	587	1	https://www.amazon.ca/MG-Chemicals-Prototyping-1-Ounce-16-Inch/dp/B008OAFOUO/ref=pd_sbs_328_2/132-5081513-4200133?_encoding=UTF8&pd_rd_i=B008OAFOUO&pd_rd_r=95480a50-

				419b-47d5-91b4-3b88102fc136&pd_rd_w=zfGQl&pd_rd_wg=FhDkE&pf_rd_p=dbebb38c-0e3d-4a67-ac15-432d7c7a2789&pf_rd_r=A5QQ0YC5SPEYDWN36ZA7&psc=1&refRID=A5QQ0YC5SPEYDWN36ZA7
5	Elegoo 120pcs Multicolored Dupont Wire 40pin Male to Female, 40pin Male to Male, 40pin Female to Female Breadboard Jumper Wires Ribbon Cables Kit for Arduino	EL-CP-004	1	https://www.amazon.ca/Elegoo-120pcs-Multicolored-Breadboard-arduino/dp/B01EV70C78/ref=sr_1_1_sspa?keywords=jumper+cables&qid=1579928593&s=industrial&sr=1-1-spons&psc=1&spLa=ZW5jcnlwdGVkUXVhbGlmaWVyPUEyVkZXSjU0U1RGSjFQJmVuY3J5cHRlZElkPUEwMDcxMTkxM0tSVVU5WVBKQVhKMiZlbmNyeXB0ZWRBZElkPUEwMzg1ODgyMUwxRTJBRE0zR1dlWCZ3aWRnZXROYW1lPXNwX2F0ZiZhY3Rpb249Y2xpY2tSZWRpcmVjdCZkb05vdExvZ0NsaWNrPXRydWU=

6	AUSTOR Lead Free Solder Wire with Rosin Core 0.6mm	AMA-17-532	1	https://www.amazon.ca/dp/B071XVPJVX/ref=sspa_dk_detail_0?psc=1&pd_rd_i=B071XVPJVX&pd_rd_w=apWS0&pf_rd_p=4b7c8c1c-293f-4b1e-a49a-8787dff31bcb&pd_rd_wg=s5ddm&pf_rd_r=Y6RJTAX1MR0X3E50MH9X&pd_rd_r=b093f5a4-5e04-4c4a-bf07-683669f8db02&spLa=ZW5jcnlwdGVkUXVhbGlmaWVyPUExWUMwVUFXVTBTMzUmZW5jcnlwdGVkSWQ9QTA0Nzk5MzAzSENLOTZDTEhKOTBBJmVuY3J5cHRlZEFkSWQ9QTAyMTk5NjJJTDlaVjVBQUtLRksmd2lkZ2V0TmFtZT1zcF9kZXRhaWwmYWN0aW9uPWNsaWNrUmVkaXJlY3QmZG9Ob3RMb2dDbGljaz10cnVl
7	Rocker Switch DPST 16A (AC) 125V Panel Mount, Snap-In	RR812C1121	1	https://www.digikey.ca/product-detail/en/e-switch/RR812C1121/EG4779-ND/2116258?utm_adgroup=Rocker%20Switches&utm_source=google&utm_medium=cpc&utm_campaign=Shopping_Switches_NEW&utm_term=&produc

				tid=2116258&gclid=Cj0KCQiAsbrxBRDpARIsAAnnz_MoJhLf6plqc9U7ZpBJv8AWE22h3Xss7RSx12FQA_XYb0KoUbjW_LYaAo6NEALw_wcB
8	M2 M3 Male-Female Nylon Hex Standoff Plastic Thread Motherboard Spacer	XHHD17041	1	https://www.amazon.ca/Male-Female-Motherboard-Prototyping-Accessories-Quadcopter/dp/B06Y4LNDH9
9	Raspberry Pi 4 Model B 2019 Quad Core 64 Bit WiFi Bluetooth (4GB)	2GB-9003	1	https://www.buyapi.ca/product/raspberry-pi-4-model-b-2gb/

3.2.5 Manufacturing PCB and Enclosure

Manufacturing refers to the process (ranger of human activities) for producing products for consumption or sale using machines, materials and tools. In essence, the raw materials are transformed into a finished good. This section of the report will outline the manufacturing process of the MAIDS project.

3.2.5.1 PCB Board Manufacturing Process

The PCB board is first designed using Fritzing software version 6.0.3. Once designed, the PCB board circuit rendition is exported for production as a RS-274X Gerber file and sent to the Humber College Prototyping Lab resulting in a professional quality DIY PCB manufactured board. The MAIDS project PCB board's design is based on a two-layer approach; separates the input and output (I/O) layer from the ground layer. While the I/O layer is located at on the top surface of the PCB board, the ground layer is located on the bottom layer. It is important to emphasize that Design Rules Check (DRC) tests are performed during the design process to ensure proper board function and reliability before the manufacturing process.

Once the design is received at the Humber College Prototyping Lab, the manufacturing process of the PCB board is subject to a few phases. To begin, the PCB board 274X Extended Gerber format file is loaded into the ProtoMat S103 LPKF plotter and router using LPKF CircuitPro software. The software converts the design from the common layout program into control data for the structuring systems, and allows for the optimization of the layout elements, and verifies design using design verification rules. Once data from the file is imported, the ProtoMat S103 circuit board plotters uses a mechanical fiducial systems (cameras for automatic position detection) {Electronics, 2020, Manual Version 0.9`, English} to assist in drilling and milling the double-sided PCB board. In the case of the MAIDS PCB board, it drills four fiducial holes using the Optical Fiducial Recognition Systems (OFRS) on the un-etched sides. This ensures that the structures on both sides of the board are matching so that they remain aligned during transfer between processes. The holes themselves use Through-hole technology; holes that go completely through the boards. In addition, the holes are non-plated (NPTH). With non-plated through holes there is no conductive path from one side of the board to the other. Connections have to be made by applying a thin wire through the hole and soldering it in place to connect the upper and bottom portions of the circuit.

Once positioning holes are drilled, the blank PCB board is transferred to the ProtoLaser S machine. The laser first creates the contours of the circuit and the proceeds to delaminate and evaporate the copper layer. The ProtoLaser S uses a laser low energy source emitting light in the green range (532 nm) {Electronics, 2008, ProtoLaser S: Operation manual 2.0`, English.}Of the visible spectrum, to systematically delaminates and evaporates the conductive copper layer from both sides of the double-sided copper clad PCB board to prevent damage to the substrate. Laser etching makes the process cost effective, fast, and robust. The result is a professional quality PCB circuit board.

Finally, the PCB board is once again transferred to the ProtoMat S103 LPKF plotter and router to cut away the laser etched PCB board from the stock board. The PCB board is then ready for the component placement and the soldering phase.

3.2.5.3 Enclosure Manufacturing Process

The enclosure is designed using CorelDraw 2018 for fast and precise laser cutting. CorelDraw 2018 can export either a .svg or .pdf landscape file which are the preferred file formats for laser cutting. The enclosure rendition document ensures that the object size is constrained within the maximum of 12″ by 24″ object size allowed on the laser cutter bed. The laser cutter uses vector lines to cut with the stroke width set to hairline (0.000 mm wide). Outside laser cuts are colored green while inside cuts are colored red. Etching of words or logos require any other color and lines thicker than 0.000 mm (hairline) which result in the burning a light layer off of the top of the material.

Once designed, the .svg or .pdf file is imported into the Trotec Speedy 100 Laser Engraver which uses JobControl laser software. It is important to emphasize that the .svg file contains not only the designed acrylic walls for the enclosure but also the acrylic joint and base designs; the Speedy 100 cuts and etches the enclosure, as well as, cut the joints to hold it together and the base board on which it rests. The laser cutter and engraver uses a 50 W CO_2 laser (Iradion tube) {Trotec, 2020, Operation Manual Trotec Job Control. Basic`, Advanced`, Expert} to cut away the enclosure and joint designs from the acrylic stock material. A fast engraving speed (4.3 m/s) {Trotec, 2020, Operation Manual Trotec Job Control. Basic`, Advanced`, Expert} creates a minimal distortion in the engraving image. Finally, separate parts are then joined together by means of acrylic joints and acrylic glue to produce the final MAIDS project enclosure.

3.2.5 4 3-D Printing: Camera Holder Bracket

The camera holder bracket manufacturing process requires 3-D printing. The design requires a SketchUp (software) produced .stl design file. The file is loaded into the Object 30 3-D printer. The Object 30 is a resin-based, ultraviolet light cured, 3-D printer. The 3-D building process begins by depositing a layer of resin 28 microns (0.0011”) thick {Geometries, 2010, Objet30 3-D Printer System: User Guide`, English. Page 6.}, and cures it (hardens it) by passing an ultraviolet light over the newly deposited resin. The 3-D printer continues the deposition process in layers with great accuracy (0.1 mm) (Geometries, 2010, Objet30 3-D Printer System: User Guide`, English. Page 6) To produce the resulting 3-D model of the camera holder bracket.

3.3 Assembly

The MAIDS project is assembled in the following stages:

1. PCB board component placement and soldering
2. Base plate assembly
3. Enclosure assembly
4. Deliverable assembly.

3.3.1 PCB Board Assembly

The assembly process begins with the PCB board. First, the connecting pins are placed onto and soldered (using safe and lead-free solder) to the laser-etched PCB board taking care that pins connectors are soldered on the appropriate layers. On the one hand, all pins connecting the main components (motion and sound sensors, and dual LED module) are soldered to the top layer of the PCB board. On the other hand, pins connected to the circuit's ground connection are soldered on the bottom layer of the PCB board. At this point, the MAIDS hardware circuit is complete.

3.3.2 Base Plate Assembly

Once the circuit is completed, the base assembly process is started. Consequently, the laser-cut base boards are glued together with acrylic glue and left to cure for at least 24 hours to assure maximum the holding power of the glue; the base serves as the physical foundation of the final product.

3.3.3 Enclosure Assembly

After the MAIDS base is finished, the enclosure assembly is completed. The parts of the enclosure that serve as side walls for the project are placed in a vice-grip for support and correct alignment. Once aligned properly, laser-cut acrylic joint plates are glued onto the walls with acrylic glue; glued walls are set aside for curing for at least 24 hours for maximum hold. After 24 hours, the enclosure is ready for complete assembly of the project.

3.3.4 Final Deliverable Assembly

To begin, the Raspberry Pi 4 Model B module is fitted, on its underside, with four M3, Male-Female, and Nylon Hex Standoffs to provide short circuit protection and adequate space clearance for the Pi from the base plate. Then, the custom made PCB circuit board is connected to the 40-pin male connector of the Raspberry Pi 4 through its own 40-pin female connector. This stage completes the hardware assembly portion of the process that serves to process signals from and control signals to MAIDS in response to intrusion events.

Once hardware assembly is finished, the assembled enclosure is placed upon the base plate to surround the hardware circuit. When aligned, the enclosure is secured to the base plate by means of two acrylic, laser-cut joint plates and acrylic glue.

Finally, the top acrylic plate of the enclosure is pressure snapped onto the top of the enclosure; the top plate serves to provide accessibility to internal components, in case of needed replacement dues to malfunction.

3.4 Tools and Facilities

3.4.1 Fabrication and Research Facilities: Humber College

The MAIDS project design, development, modification and production was possible by the use of the following lab and library facilities located at the North Campus:

1. The Humber prototype lab facilities (Rapid Prototyping lab) located in building J, Room J201.
2. The Advanced Electronics Lab located in building J, Room J232.
3. The Humber College Library, located on the 4th floor of the Learning Resource Commons was used during the project research phase.

3.4.2 PCB Board Cutting and Etching

The PCB board design using Fritzing software and produced a 274X Extended Gerber format file employed by the ProtoMat S103 LPKF. The ProtoMat S103 is a circuit board plotter for Contour routing of the circuit board (double layer copper foil) and an Optical fiducial recognition systems for automatic position detection to assist in drilling and milling of double-sided PCBs. It has a spindle speed of 100,000 rpm, a maximum travel speed of 150 mm/s, repeatability of ± 0.001 mm (± 0.04 mil), drilling speed of 120 strokes/min and a resolution of 0.5 µm (0.02 mil). {Electronics, 2020, Manual Version 0.9`, English}The lead time for the PCB board is 12 hours.

PCB board etching is accomplished by the ProtoLaser S which etches away the copper from the double-sided copper clad PCB board. The ProtoLaser S has a minimum track/gap of 50 µm/25 µm (2 mil/1 mil), a resolution scan field of 2 µm (0.08 mil), a laser pulse frequency 10 - 100 kHz, continuous wave (CW), and cutting bed dimensions of (12“ x 24“) {Electronics, 2008, ProtoLaser S: Operation manual 2.0`, English.}

3.4.3 Laser Engraving

Trotec Speedy 100 Laser Engraver is used for acrylic enclosure cutting and etching. The laser engraver requires a design in .pdf format. The Speedy 100 uses a 50 W CO_2 laser (Iradion tube), uses 2 inch lens (standard), a cutting speed of 180 cm/sec speed and its bed size can accommodate objects of up to 12" x 24". (Trotec, 2020, Operation Manual Trotec Job Control. Basic`, Advanced`, Expert)

3.4.4 3-D Printing

3-D printing requires a SketchUp produced .stl design file which is then created on the Object 30 3-D printer. The Object 30 is a resin-based, ultraviolet light cured, 3-D printer with Support Material SUP705 (WaterJet removable) and SUP706B (soluble), maximum build size of 294 x 192 x 148.6 mm (11.57” x 7.55” x 5.85”), layer thickness of 28 microns (0.0011”) and 16 microns (0.0006”) for VeroClear material, accuracy of 0.1 mm (0.0039”) and employs a fusion deposition system. {Geometries, 2010, Objet30 3-D Printer System: User Guide`, English. Page 6.}

3.4.5 Soldering

ESD safe, Small footprint (5.9” X 4.5” X 3.6”) Weller WESD51 Soldering Station with power consumption of 50 watts, temperature range 350 ºF – 850 ºF, operating voltage (output) 24 Volts, Temperature Stability +/-10°F (6°C) and Heating Element Type Nichrome Wound; Fiberglass and ceramic insulated. {Elektrotanya, 2020, Model WES51 Electronic Soldering Station}

3.5 Shipping, duties and taxes

When shipping finished goods domestically or internationally, it becomes extremely important to consider the effects of shipping charges, duties, and taxes. Shipping internationally can help a business grow financially, in reach and reputation. However, not understanding shipping taxes and duty costs can create massive headaches for the business. Every country has its own laws and rates, and businesses from different sectors face different compliance challenges when shipping internationally or domestically. Depending on the shipment content and the destination, charges could significantly impact the total shipment cost and the products end price. This section presents the shipping, duty and tax charges incurred during the development of the MAIDS system, along with other costs. It is presented in table form for clarity, in explicit order and includes the following sections: Description, quantity, Unit price, Discount, Federal Tax [GST/HST/TPS/TVH], Provincial Tax [PST/RST/QST/TVP/TVD/TVQ], Shipping charges and total cost. The MAIDS shipping, duties and taxes applicable to each component or part is found in the table below.

Table 2 Per Component Costs, Quantities, Discounts, Shipping, Duties And Taxes.

Description	Quantity	Unit Price	Discount	Federal Tax	Provincial Tax	Shipping Charges	Total Cost
Sound Sensor Module	1	$10.30	-$1.10	$0.00	$0.00	$3.99	$13.19
Active Buzzer Module	1	$7.99	-$0.18	$0.00	$0.00	$0.18	$7.99
Motion Sensor	1	$10.99	$0.00	$0.00	$0.00	$0.00	$10.99
Raspberry Pi 4 Fan	1	$9.99	$0.00	$0.00	$0.00	$0.00	$9.99
LED Module	1	$12.98	$0.00	$0.00	$0.00	$0.00	$12.98
SanDisk Ultra 32GB SDHC Card	1	$13.99	$0.00	$0.00	$0.00	$0.00	$13.99
Power Supply RPI4	1	$13.59	-$6.99	$0.00	$0.00	$6.00	$13.59
Acrylic Sheet	1	$13.78	-$6.99	$0.00	$0.00	$6.99	$13.78

3.6 Working time versus lead time

A better understanding of lead time (LT) can lead to very substantial gains. For example, a company can develop a more profitable scheme that can meet customer requirement more efficiently. In addition, a greater understanding of LT leads to the detection and correction of performance issues that can be corrected quickly. Finally, it leads to the improvement of customer relations by increasing the level of communication. {Rajaniemi, 2012, Measuring and Defining Lead Time in a Telecommunication Production}

It is important to emphasize that there was no stipulated time limits on the number of hours one was required to work on the MAIDS project on the part of Instructors or Humber College, other than, completion of project by the course's schedule end date. The working time schedule adopted was a voluntarily established and followed project schedule.

The order lead time, the time from order received to customer order delivered was approximately 8 weeks. The order handling time, the time from customer order received to sales order created was less than 12 hours. The manufacturing lead time, Time from sales order created to production finished (ready for delivery) was approximately 2 weeks. The production Lead Time - Time from start of physical production of first submodule/part to production finished (ready for delivery) was approximately 2 weeks. The Delivery Lead Time - Time from production finished to customer order delivered was approximately 1 week due to testing and slight software modifications. {Rajaniemi, 2012, Measuring and Defining Lead Time in a Telecommunication Production}

The working time, time to assemble the complete MAIDS deliverable, was 2.0 hours per work day, five days per week (total of 10 hours per week), for a total of 20 hours, total.

4.0 Development Platform

4.1 Mobile Application

MAIDS uses an Android-based application written in the Java language and built on the Android Studio IDE version 3.5.3 to control and test components, as well as, display intrusion related information. Powered by Gradle, Android Studio's build system allows for a customized MAIDS build and generates multiple build variants for different devices from a single project. In the case of the MAIDS project, the two variants devices created are:

1. A 10" display generic tablet running a quad-core CPU (Central Processing Unit) with 2 GB of RAM, using Android version 6.0 at a resolution of 1536 x 2048 pixels.
2. A 6" display Google Nexus 6 running a quad-core CPU, with 3 GB RAM, using Android version 6.0 at a resolution of 1440 x 2560 pixels.

In addition, the emulator permits virtual testing of the builds, simulates different MAIDS configurations and features, and provides feedback on feature response and configuration performance used to quickly modify the application.

Building the Android-based MAIDS application requires the use of three distinct file types:

1. Manifest
2. Activity
3. Drawable resource.

The Android manifest file is named AndroidManifest.xml and must be included with every application in the root directory. The file contains essential application metadata, a set of data that describes and gives information about other data, in Extensible Markup Language (XML) format (a textual data format with strong support via Unicode for different languages). The manifest file presents essential information about the application to the Android system, information the system must have in order to run any of the application's code. Specifically, the MAIDS manifest file contains the basic building blocks of application (i.e. activities, services, permissions, etc.), details about resource permissions (i.e. access the Internet, remote storage devices, etc.) and the set of classes needed before launch.

Generally, an Android activity file refers to one screen of the Android application’s user interface (more commonly referred to as the Application Program Interface (API)) and may contain one or more activities (screens); the main activity is shown first when the application starts. Subsequent screens, require their own activity. Specific to MAIDS, the two pertinent activities will be discussed below in a more detailed fashion under their own section of this report. Each section will outline a general description of its operation mode, as well as, detailed information about the inner workings of the program’s code.

Finally, MAIDS uses a drawable resource file which is a graphic file (i.e. .png, .jpg, .gif) that can be drawn to the screen. Our MAIDS application uses exactly two .jpg format files on the second activity; one is used as a placeholder for the intrusion photo while the remote photo of the intrusion is retrieved and the other is the intrusion photo itself, once retrieved. The intrusion photo will have a caption which contains the following information:

1. MAIDS Alarm System Header
2. Address of Intrusion
3. Room where intrusion took place

4. Date and time of intrusion.

The Android application uses a landscape layout (relative layout to be more specific) on the tablet and a portrait layout for the phone, for better visualization of component views and is divided into two main activities:

1. A login activity (first screen), which allows for user authentication and access to the network
2. An data visualization display and component control activity (second screen) which relays database intrusion data (i.e. intrusion date and time, place of entry, owner contact information, etc.), displays real-time statistical information about the Raspberry Pi system, as well as, a remote control mechanisms to activate and deactivate MAIDS and test some of its components (LED module, in particular).

4.2 Login Activity

The login activity is the first screen the user encounters to access MAIDS remotely. The main class of the program is: MainActivity. The class is responsible for presenting the user with a welcoming screen and a MAIDS promotional advertisement photo. In addition, the MainActivity class displays two rectangular, labelled (username and password), user input textboxes used for authentication purposes. Moreover, the class displays two labelled buttons (login and cancel) used to either login into the system and access the remote control features of the application or cancel access to it. It is worth noting that the application has been coded to allow the user only three tries at authentication; otherwise, the application closes and the login process has to be restarted. Once the user inputs the correct username and password, control is transferred to the second screen, MainActivity3.

Programmatically, the main activity consists of a public class named MainActivity. Inside the class, there is the protected onCreate() method containing two button views (login (b1) and cancel login (b2)), two EditText views (user input boxes ed1 and ed2) and one text view (tx1) which displays a red horizontal bar and are arranged in a relative layout. When the user is presented with the initial screen s/he has the option of login into the application or cancel the login. If the user chooses to launch the application s/he must first enter two pieces of information, username and password, into the textboxes available on the screen. When clicking the login button, the activity activates the setOnClickKistener(v) method which retrieves the EditText box inputs and checks the username and password imputed against the programmed login settings. If they match, the application will close the login activity screen and display the second activity screen which contains the data display activity and control activity on the same screen. At the same time the login button is clicked, a Toast message is display (“Redirecting…”) at the bottom of the screen through which the user is informed of the subsequent activity to be displayed (MainActivity3.javaj). The user has three opportunities to launch the application with each try displayed in the form of a horizontal red line increasing in length referencing the number of tries. If, upon the third try, the user does not input the correct username and password authentication fails and the application textbox inputs are cleared and the user is asked to re-enter the information.

4.3 Data Visualization Activity

The purpose of the data visualization activity is to present MAIDS-created intrusion data in a visual, easy-to-read-and-see format to the user. The data visualization activity is incorporated within the second activity (or screen). The second screen displays five, sequentially arranged, gray, rectangular buttons (views) situated on the lower portion of the screen. The button are labelled as follows:

1. Retrieve Intrusion Database
2. Retrieve RasPi Stats
3. Activate Maids
4. Deactivate Maids
5. Test Led Module.

Each button is a subclass onto itself performing different local and remote activities such as: data visualization, and action control. The main data visualization elements (views=buttons) for the MAIDS application are labelled: retrieve intrusion information and display real-time MAIDS system statistics locally. The main action control elements (views) are labelled: retrieve statistics from server, activate MAIDS, deactivate MAIDS, and test LED module.

Programmatically, the data visualization activity (Main3Activity class) is contained inside the Main3Activiy.java file of the application. The data visualization portion of the activity consists of two WebView elements (htmlWebView) and five buttons. The first WebView element displays the database information gathered form MAIDS internal server (and reached via the Internet through https://singular-gar-5555.dataplicity.io/maidsintrusion.php link). The second WebView element displays real-time statistical information of the Raspberry Pi 4. The two buttons (mButton1 and mButton3) which activate their respective setOnClickListener (v) method to retrieve the database and statistical information remotely. All these view elements are located inside the onCreate () method of the Main3Activity. It is important to emphasize that during the database and statistical information retrieval process, Toast messages (messages that provides simple feedback about an operation in a small popup) such as “Retrieving DB information…” and “Retrieving RASPI Stats…” are displayed to the user informing them of the action being conducted.

4.4 Action control activity

There are four action control elements to the MAIDS project:

1. Test LED module
2. Retrieve Real-Time Raspberry Pi 4 Statistics information
3. Activate MAIDS remotely
4. Deactivate Maids remotely.

All of these actions are contained inside the onCreate () method of the Main3Activity class and ran through their respective setOnClickListener (v) method.

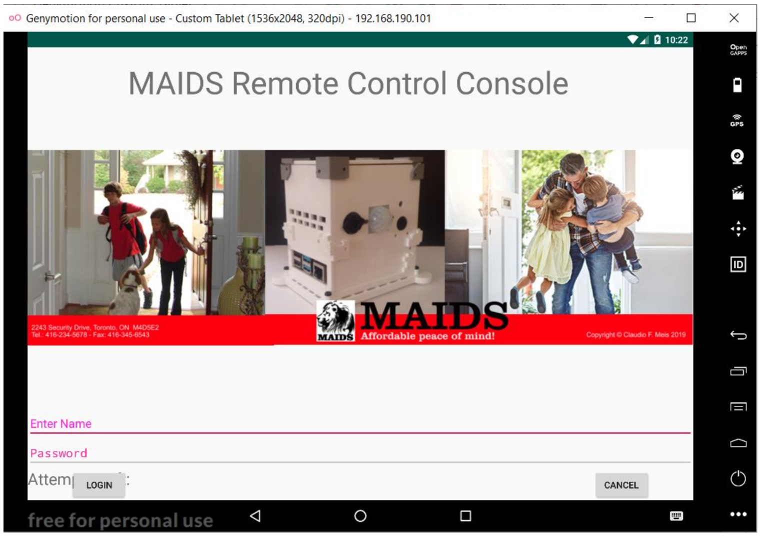
When the button named mButton3 is clicked, the attached listener method displays a Toast message ("Testing LED module…") and call upon the testLedsCommand () method. The method takes as input the username, password and host strings, as well as, a port integer value. Using these parameters, the method connects via SSH (inside the jsch library) to the MAIDS device and runs an internal python v3.0 program (python3 testleds.py) to test the LED module remotely. The test is programmed to assess the function of the green and red LED lights of the MAIDS device through two output GPIO pins and to intermittently turn them ON and OFF five times for a period of 2 seconds, each.

When the button named mButton2 is clicked, the attached listener method displays a Toast message ("Retrieving RASPI Stats …") and retrieves real-time statistical information from the MAIDS internal server. The attached listener method to mButton2 loads the URL http://192.168.0.19:8080 to display the real-time statistical information of the MAIDS device. Each step of the session creation, session connection, channel connection and the downloading of the remote file is presented to the user in the form of Toast messages at the bottom of the screen.

When the button named mButton4 is clicked, the attached listener method displays a Toast message (“Activating MAIDS remotely…”) and calls upon the maidsOnCommand () method. The method takes as input the username, password and host strings, as well as, the port integer value. Using these parameters, the method connects via SSH (using the JSCH library) to the MAIDS device and runs through the ‘exec’ command an internal python v3.0 program (python3 maids_final_python_code_22102019_bak1.py) that initializes GPIO pins, sounds audible warnings and activates the MAIDS alarm system. Each step of the session creation, session connection, channel connection and the downloading of the remote file is presented to the user in the form of Toast messages at the bottom of the screen.

When the button named mButton5 is clicked, the attached listener method displays a Toast message (“Deactivating MAIDS remotely…”) and calls upon the maidsOffCommand () method. The method takes as input the username, password and host strings, as well as, the port integer value. Using these parameters, the method connects via SSH (using the JSCH library) to the MAIDS device and runs through the ‘exec’ command an internal python v3.0 program (reboot.py) that reboots the MAIDS alarm system. Each step of the session creation, session connection, channel connection and the downloading of the remote file is presented to the user in the form of Toast messages at the bottom of the screen.

4.5 Testing Screen Shots of MAIDS Android Application

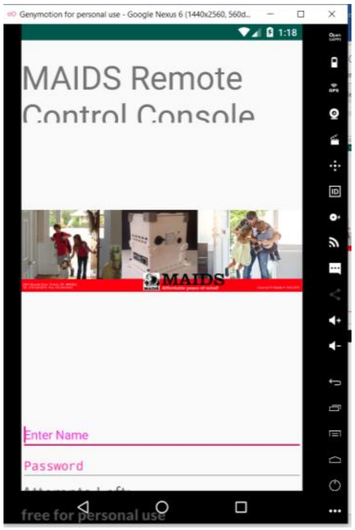


Genymotion for personal use - Custom Tablet (1536x2048, 320dpi) - 192.168.190.101

MAIDS Remote Control Console

10:22
MAIDS
Affordable peace of mind!
2243 Security Drive, Toronto, ON M4D5E2
Tel: 416-234-5678 - Fax: 416-345-6543
Copyright © Claudio F. Meis 2019
Enter Name
Password
Attem
LOGIN
CANCEL
free for personal use

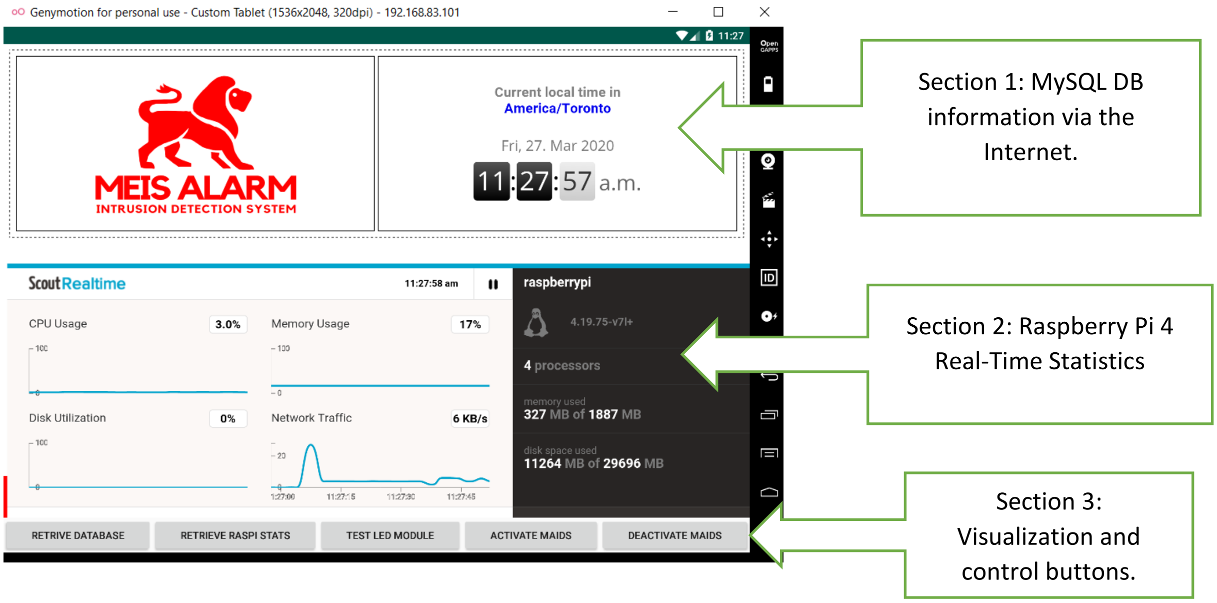
FIGURE 1 MAIDS LOGIN ACTIVITY TABLET SCREEN.



Genymotion for personal use - Google Nexus 6 (1440x2560, 560d...

MAIDS Remote Control Console

FIGURE 2 MOBILE ACTIVITY SCREEN.



Genymotion for personal use - Custom Tablet (1536x2048, 320dpi) - 192.168.83.101

Current local time in America/Toronto

Fri, 27. Mar 2020

11:27:57 a.m.

ScoutRealtime 11:27:58 am

CPU Usage 3.0%

Memory Usage 17%

Disk Utilization 0%

Network Traffic 6 KB/s

raspberrypi

4.19.75-v7l+

4 processors

memory used 327 MB of 1887 MB

disk space used 11264 MB of 29696 MB

Section 1: MySQL DB information via the Internet.

Section 2: Raspberry Pi 4 Real-Time Statistics

Section 3: Visualization and control buttons.

FIGURE 3 VISUALIZATION AND CONTROL ACTIVITY SCREEN.

4.6 Android Application Status Report

Prepared by Claudio F. Meis, January 31, 2020.

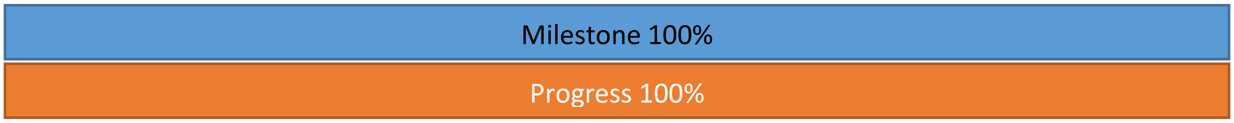
The presentation of the MAIDS project at the Capstone Project EXPO at 1:00 – 4:00 p.m. on Thursday, April 9, 2020, is still on track.

The following work has been completed on the MAIDS Android-based application:

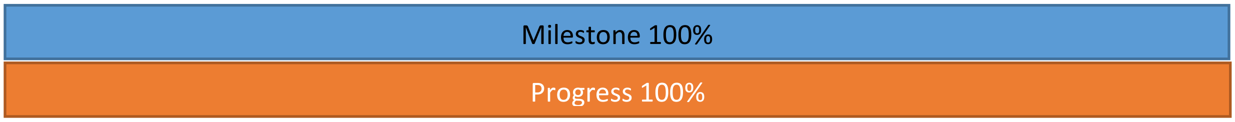
- Coding for the login activity.
- Coding for the database information retrieval.
- Coding for the real-time statistical information of MAIDS
- Coding for the remote testing of the LED module.
- Coding for the remote activation of MAIDS device.
- Coding to remotely turn OFF the MAIDS device.

Progress against Milestones

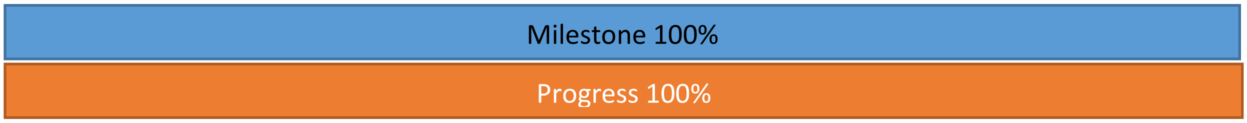
Login Activity



Data Visualization Activity



Control Activity



Key Issues

- None.

Action Steps

- None

5.0 Image/firmware

5.1 Image Creation Process for Raspbian OS on 32 GB SCHD Card

The SCHD card that was used for MAIDS did not come with the Raspbian operating system installed on it. In order to install the Raspbian OS image onto the card, the NOOBS and balenaEtcher software are used. To begin, the NOOBS zip archive is downloaded from https://www.raspberrypi.org/downloads/noobs/. Once NOOBS has been downloaded, the SCHD card must be formatted. The formatting of the card is done with balenaEtcher which is a free and open-source utility used for writing image files onto storage media to create live SD cards and USB flash drives. Next, the files from the NOOBS zip archive are extracted onto a directory. Finally, all the files are selected and dragged onto the SCHD card. Once the files have been copied over, the card can be ejected and used on the MAIDS device.

5.2 Firmware General Description and Requirements

The MAIDS projects incorporates its own custom-made firmware to run and control the functionality of the alarm system. Firmware refers to the firmware program, composed of individual instructions that are programmed onto the hardware device. In MAIDS case, the firmware provides the needed instructions for the computer hardware to communicate and control the PCB board components; it is stored in the 32 GB SCHD card of the device.

The development of the firmware takes into account the need to meet the real-world requirements of the project:

1. Provide a visual alarm through the dual LED module.
2. Provide an audible alert through audible warning and intrusion messages.
3. Provide rapid-response multi-channel alarm notifications (Email w/ photo, push notification, SMS messages and phone calls) in real-time.
4. Provide sound detection for gunfire, loud voices and/or breaking of glass.
5. Provide a remote method for testing visual alarm component functionality
6. Provide Internet-based database (MySQL) information for authorities use for prosecution.

Moreover, the firmware is implemented via version 3.6 of the Pyhton programming language. Python is a multi-purpose, high-level, interpreted programming language. The MAIDS project's firmware requires the following minimum requirements to run: an Intel Atom® processor or higher, at least one gigabyte of disk space, two gigabytes of RAM (recommended), the Raspbian (Buster) operating system (kernel version 4.19) and a Linux- 64-bit x86 system architecture.

The hardware’s inter-component interaction (Raspberry Pi and PCB board sensors) is mediated via the RPI.GPIO library. The GPIO (general-purpose input/output) pins on a Raspberry Pi (with a 40-pin GPIO header) interfaces with the physical devices of the PCB board (i.e. button, LED module, motion sensor, sound sensor, etc.). The RPi.GPIO library handles the interface with these pins and allows the user, programmatically, to implement their function and control the device at run time.

5.3 Firmware and Hardware Interaction Description

The following block diagram demonstrates the general hardware and firmware interaction of the MAIDS project; its operative functionality is fully detailed below.

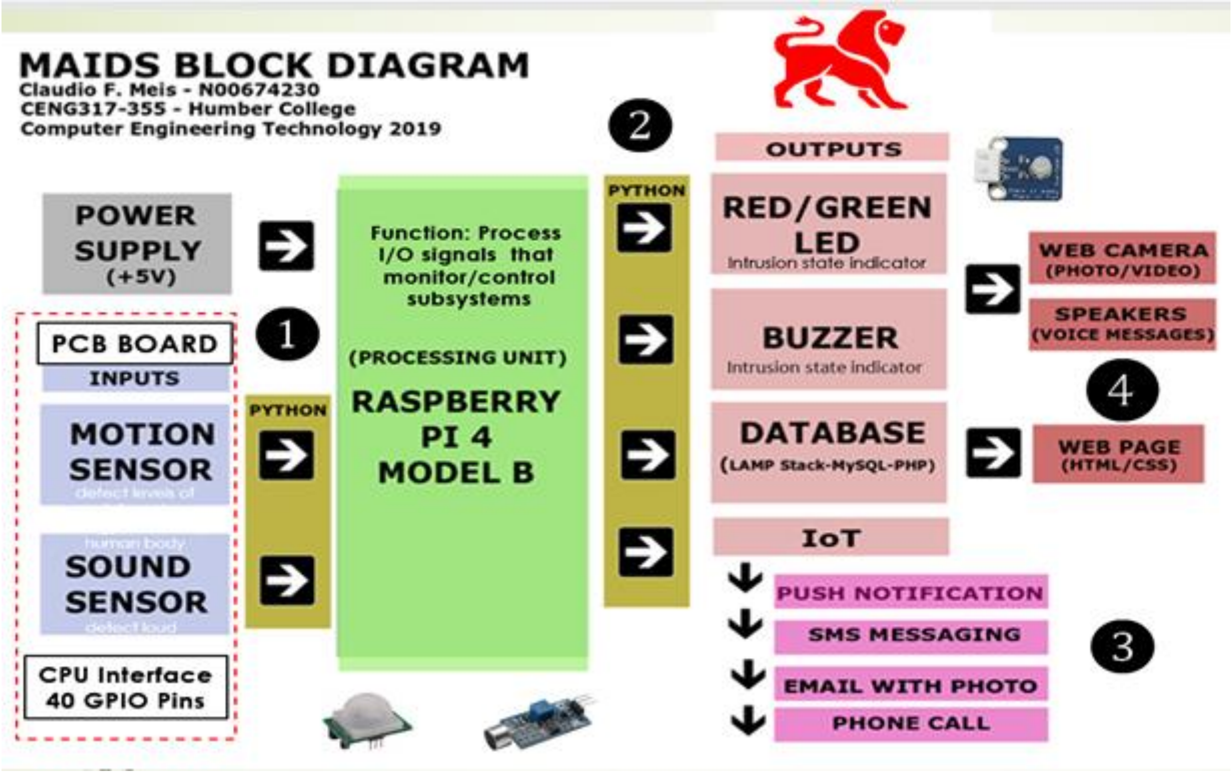


FIGURE 4 BLOCK DIAGRAM OF HARDWARE AND FIRMWARE FOR MAIDS.

In collaboration, MAIDS hardware and firmware work as follows:

- Start of section ❶ of firmware. Pressing the rocking switch to the ON position powers and activates the MAIDS system, and generates a signal to the green-LED output GPIO pin.
- Then, the firmware runs the welcoming screen and pertinent MAIDS system information function.

- Thereafter, the program displays the message: “MAIDS Surveillance Mode - Arming...” Once the system is armed, it warns the user by means of an audible message to clear the room and a countdown commences (10 down to 1).
- Upon conclusion of the countdown, the program displays the message: “MAIDS Surveillance Mode - Armed and Active...” At this point, MAIDS is activated and actively surveilling the space around it.
- Start of section ❷ of firmware. If motion or sound is detected by the motion (PIR) or sound sensors on the MAIDS PCB board, they will generate an input signal. The motion/sound input signal travel through the connecting wires to the GPIO pins (setup with the BOARD numbering system) to the Raspberry Pi 4 Model B and the firmware processes the input signal on the particular input GPIO pin and provides the following response outputs: (Start of section ❸ of firmware)

    ii. An audible message plays stating that either a motion or sound intrusion has been detected, that it is unlawful and that authorities are being notified.

    iii. The green LED then turns OFF and the red LED light is turned ON, after which it will flash intermittently ON/OFF five times.

    iv. Then, the message is displayed: “Sending push notification to Android phone...” After the message, via the Internet, the Pushover server is reached and originates the push notification to the user’s Android phone.

    v. Afterwards, another message is displayed: “Sending SMS notification to Android phone...”; again via the Internet, a Twilio (cloud communications platform as a service company) SMS server is reached and a SMS message is generated and sent to the user’s Android phone along with a confirmation message that is displayed on the screen (i.e. “Message sent ID: SM79c3534ddeb648ee898b7d4a0e76872f”)

vi. The firmware then displays the message “Sending Intrusion Alert to email...” and initiates the following sequence of events:

- Logs into the user’s email server with the user’s username and password.
- Runs a program named ‘fswebcam’ to take a picture of the intruder and adds intrusion information (time, place, address and alert message) onto the picture’s caption.
- Then, generates the email with picture which is then sent, along with an intrusion alert message, to the user’s email.
- At this point, the message “Calling Android phone...” is displayed and a phone called is placed to the user via Twilio server; a confirmation of the call made is displayed on the screen (i.e. “Phone call ID: CA3c525429de7a5145939b559285c96095”

vii. Start of section ❹ of firmware. Immediately after the email is sent, the message “Appending Intrusion Alert to Database...” is displayed and the following takes place:

- A record of the intrusion stating intrusion id, time, place, person to contact, contact’s phone number and email is generated and saved onto the following databases:
    - MySQL Text-based database
    - MySQL GUI-based database (entries accessible via browser)

- Once the entry is recorded in the database, the program displays the following message: “Record inserted successfully into MAIDS-DB; MySQL connection – CLOSED”

viii. Finally, a siren audible is blasted through the connected speakers to frighten the intruder.

5.4 Firmware’s Functional Design

Programmatically, MAIDS firmware is composed of fourteen Python functions. A function is a block of code that only runs when it is called. Two of the functions (motion and sound) require input parameters, the rest, do not. The firmware includes:

1. The set_up_gpio() function which sets up the GPIO numbering system, input and output pins and disables warnings.
2. The maids() function which displays a welcome screen and firmware information (i.e. license, author, version, etc.).
3. The notify() function which updates the database, sends email and sends push notification to android phone.
4. The alarm() function which sets the led flashing pattern.
5. The alert() function which flashes a visual alarm, plays an audible intrusion message and sounds the alarm audible.
6. The sound(sound) function which sets up the sound sensor’s functionality.
7. The motion(motion) function which sets up the motion sensor’s functionality.
8. The send_mail() function which sends an email alert with picture to recipient.
9. The send_androidpush() function which sets up the android push notification information with the pushover service.
10. The appendtodb() function which appends the intrusion data to a text database.
11. The intruderwarning() function which plays an intrusion warning message.
12. The siren() function which plays a siren alarm sound.
13. The activationwarning() function which sets up an audible activation countdown.

14. The dbinsert() function which inserts the intrusion information into the MySQL database.

Furthermore, the functionality of the firmware code is derived through the use of the following Python modules:

1. RPi.GPIO
2. time and datetime
3. smtplib
4. ssl
5. os
6. http.client
7. urllib
8. vlc
9. mysql.connector
10. email (with specific module imports email.mime.multipart, email.mime.base, email.mime.text and email.utils)
11. subprocess

No detailed description of each module’s functionality is presented due to their lengthy nature. However, quick online searches can easily provide this information, if needed.

Finally, database creation and implementation required the use of interpolated MySQL programming commands (i.e. SELECT, INSERT, CREATE TABLE, etc.).

5.5 Code runs via CLI or remote desktop

Remote desktop is a firmware feature which permits users to connect to a remote computer, meaning a computer that is situated in a different geographic location, and interact with it as if the computer is located in front of the user. People use remote desktop access for an assortment of tasks, including: access a workplace computer from home or while roaming, access a home computer from any location, fix computer problems, perform administrative tasks or demonstrate a process or a software application.

In regards to MAIDS, one can connect to the device either locally or remotely, be it from another part of the same room or building, or from halfway around the world. As long as the MAIDS device is connected to the same network or to the Internet, the code for the MAIDS project can be run either through the Raspberry Pi command line interface (CLI) or through a remote desktop. In MAIDS case, remote desktop access refers to the secure access of the MAIDS device through the Android application running on an Android phone, tablet or through the execution of the Python source code from the CLI of another remote computer.

5.6 Wireless connectivity

Wireless connectivity is achieved via configuration of the wpa_supplicant.conf file. This file is created manually as ‘root’ and saved in the /etc/wpa_supplicant directory. The wpa_supplicant file is configured using a simple text file that specifies all accepted networks and security policies, including pre-shared keys. The file manages the configuration and administration of wireless networks on Raspbian Linux OS so that the MAIDS device can connect to configured and available wireless network. The wpa_supplicant “… implements WPA key negotiation with a WPA Authenticator and EAP authentication with Authentication Server. In addition, it controls the roaming and IEEE 802.11 authentication/association of the wireless LAN driver.” {Malinen, 2020, wpa_supplicant(8) - Linux man page}It runs as a background daemon that controls the wireless connection. The wpa_supplicant “…automatically selects the best network based on the order of network blocks in the configuration file, network security level (WPA/WPA2 is preferred), and signal strength.” {SysTutorials, 2019, wpa_supplicant.conf (5) - Linux Man Pages}The MAIDS project makes use of Wi-Fi Protected Access (WPA, also known as the IEEE 802.11i-2004 standard.) which is a type of encryption (protocols and security certifications) employed to secure the mainstream of Wi-Fi networks; a unique encryptions key (Pre-Shared Key that is 64 hexadecimal digits long) is used in WPA for every wireless client accessing the network. The main purpose for the use of encryption is to protect the confidentiality of data and to assist in the protection of its integrity. In addition, it is worth noting that WPA2 negligibly impacts network performance because of the extra processing load of encryption and decryption during every connection.

The network configuration (of the wpa_supplicant.conf file) for the MAIDS project is presented fully in Appendix II of this document and found on the GitHub repository with link:

1. https://github.com/srgawain2264/CENG317-MAIDS_PROJECT/blob/master/CENG355/wpa_supplicant.conf

In the MAIDS case, actual wireless connectivity is accomplished through the use of the RealVNC software. Real Virtual Network Computing (RealVNC) is a desktop sharing system that allows for the remote control of another computer; it can be downloaded from https://www.realvnc.com/en/connect/download/vnc. RealVNC works by transmitting all keyboard and mouse movements from one computer to another. All that is required to run RealVNC is a network TCP/IP connection, a VNC server (Installed on the MAIDS device), and a RealVNC viewer (installed on any other device used to communicate with the MAIDS device, i.e. Android tablet or phone) in order to connect to the computer that is running the server. The RealVNC Server application must be running on the device that is being controlled (MAIDS) before a connection is attempted. Once the RealVNC Server is running on the computer to be controlled, it will appear as a selectable option in RealVNC Viewer. A login screen will appear on the device connecting to the server and then the login information (for the MAIDS device) is introduced to log into MAIDS. Once the password is validated, the desktop of the MAIDS device is shown in the RealVNC Viewer of the connecting computer, phone or tablet.

5.7 Sensor/effector code on repository

The Python source code for the MAIDS project is included in Appendix I of this document. The code file (maids_final_python_code_ceng355.py) is also found online through the MAIDS GitHub repository link:

1. https://github.com/srgawain2264/CENG317-MAIDS_PROJECT/blob/master/Python%20Code/maids_final_python_code_ceng355.py.

In order to maintain the project current, any and all future modifications to the code are promptly updated to the GitHub site.

6.0 Breadboard/Independent PCB

6.1 MAIDS Schematics

Development of the MAIDS breadboard prototype and the custom-made PCB board began with the schematics diagram. The schematic will serve as a blueprint for laying out the traces and placing the components on the PCB board. The MAIDS schematic diagram (also called wiring diagrams or circuit diagrams) is a representation of the significant components of the system using standardized symbols and lines. That is to say, its schematics show how the different components (LED module, sound and motion sensors) of the circuit are connected. In the schematic diagram, lines represent connecting wires, while other elements like the LED module, and the motion and sound sensors are represented by standardized symbols called electrical schematic symbols. In addition, is it worth noting that the MAIDS schematic diagrams is useful to explain the general way that its electronic system works.

MAIDS' schematic diagram was developed using the University of Applied Sciences Potsdam, English language, Fritzing software version 0.9.4. Fritzing is an open-source CAD (Computer Aided Design) software used in the design of electronics hardware, breadboard prototypes and PCB board circuits. Fritzing installation requires one of the following operating systems:

1. Windows 10 (Windows 7 is reported to work, too)
2. Mac - OSX 10.14 and up, though 10.13 might work too.
3. Linux - a fairly recent Linux distro with libc >= 2.6

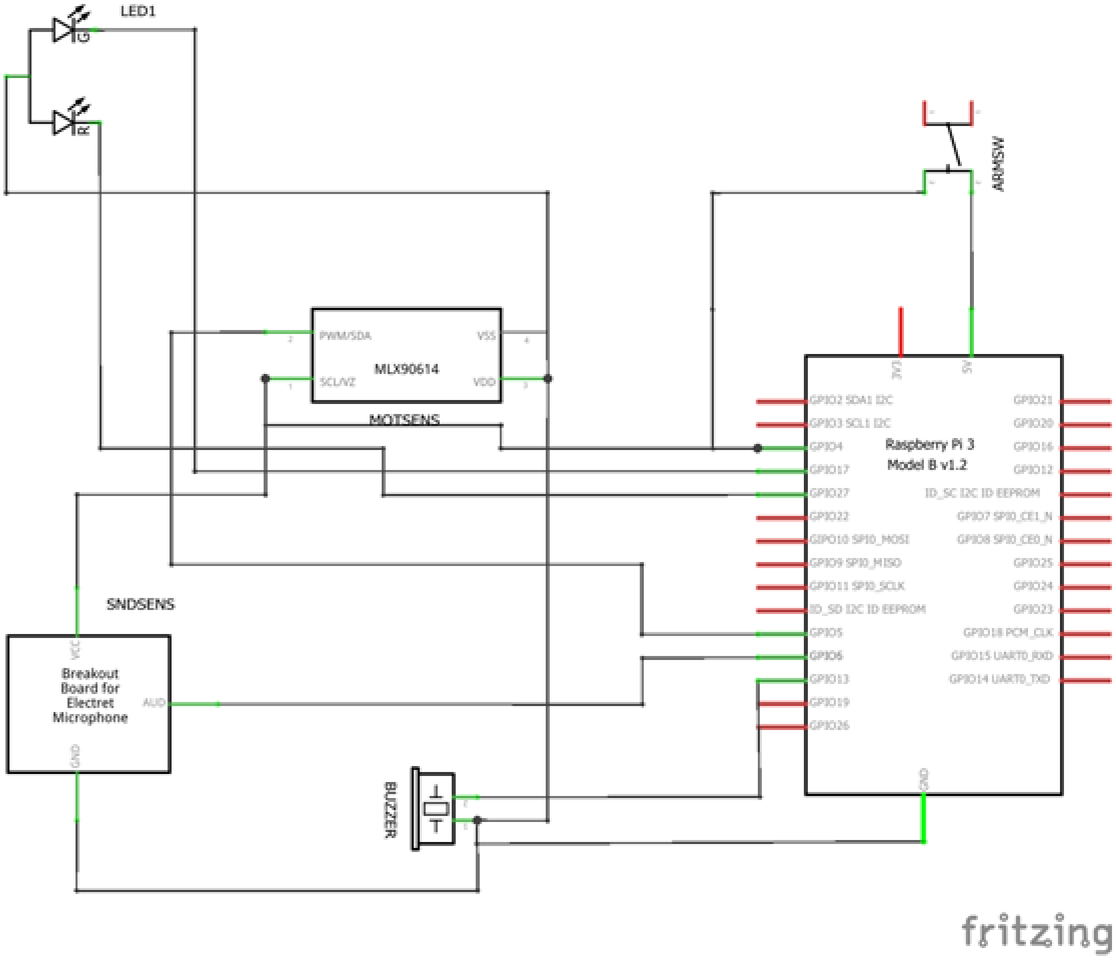


FIGURE 5 MAIDS SCHEMATIC DIAGRAM.

Essentially, the schematic design consisted of dragging and dropping the necessary components from the Fritzing built-in libraries onto the schematics. Once all of the symbols are placed on the schematic and the footprints to each symbol has been assigned, the wires are drawn connecting the components together.

6.2 MAIDS Breadboard

A solderless breadboard is an invaluable aid to prototyping a circuit for a project. The term breadboard originated during the vacuum tube era in the early 1920s. Tubes were plugged into sockets. The sockets and other large components were then screwed or nailed to wooden boards used for rolling dough. These breadboards made an ideal mounting platform for the components, and gave birth to the technique's name. Interconnections were made by soldering wires between appropriate pins on the tube sockets. Power and ground buses—made from heavy copper wire—were nailed or screwed to the wooden board. Early breadboards often used additional nails as connection points where wires could be wrapped and soldered. Terminal strips were also used for interconnection points. Nowadays, solderless (do not require soldering to make connections) breadboards are the norm.

The breadboard used in the MAIDS project is a full size board made from plastic and is rectangular in shape. In this typical solderless breadboard, the holes are designed to accept standard IC pins on 0.1" centers. Internally, the center part of the board is divided into two rows that are subdivided into a number of vertical columns having five pins connected together. The two horizontally connected rows at the top and bottom of the board make convenient buses for supply voltages and ground connections.

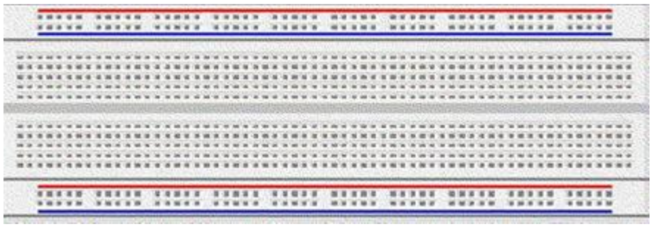


FIGURE 6 TYPICAL PROTOTYPING BREADBOARD (KESTER, 2016)

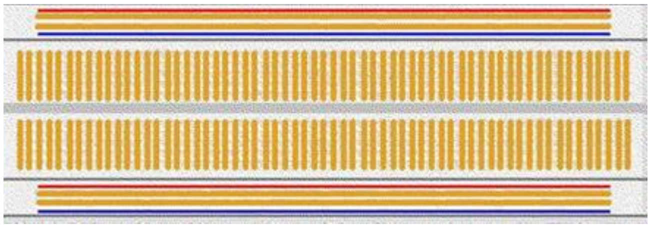


FIGURE 7 BREADBOARD INTERNAL CONNECTIONS (KESTER, 2016)

The leads of components such as LEDS module, sound and motion sensors, are inserted into the holes. Each set of five holes connected by a metal strip underneath forms a node (a point in a circuit where two or more components are connected). Connections between the different MAIDS components are made by inserting their leads in a common node. The long top and bottom row of holes, indicated by the red and blue stripes are used for power supply connections (3.3 V DC and ground (GND)). The rest of the circuit was built by inserting components and connecting them together with jumper wires. It is worth noting that solid core wires rather than stranded wire are best to use with solderless breadboards. The connections are not permanent, so it is easy to remove components.

The resulting MAIDS circuit breadboard prototypes (Fritzing and actual) are shown below.

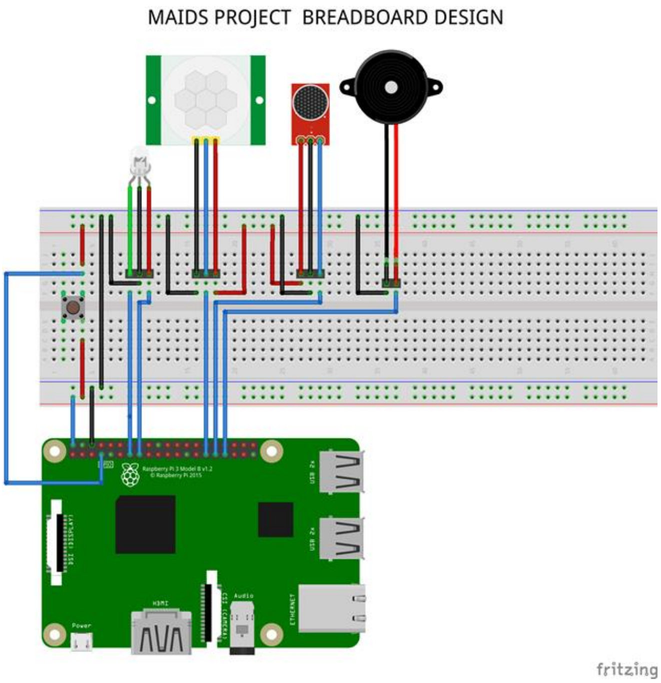


FIGURE 8 BREADBOARD CIRCUIT PROTOTYPE IN FRITZING.

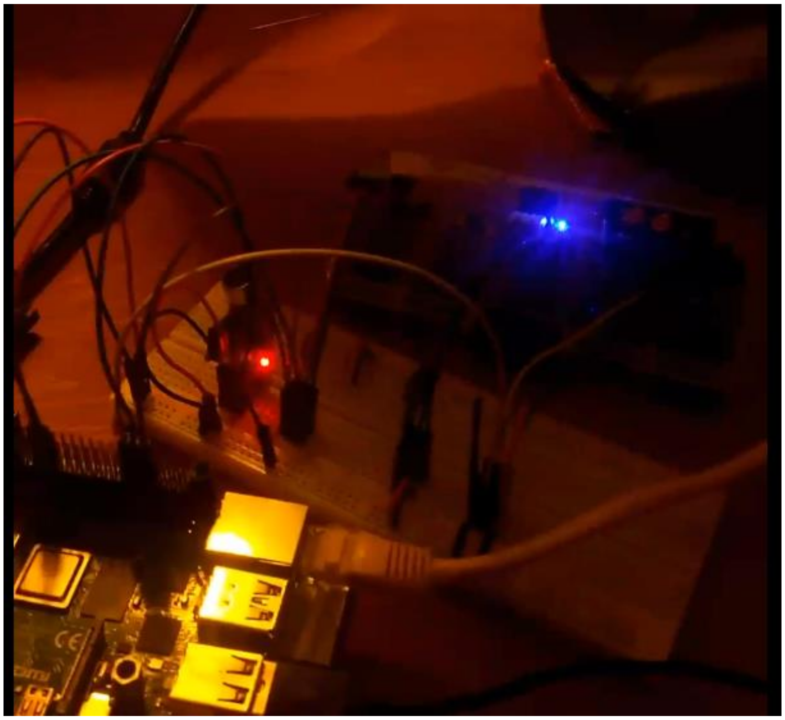


FIGURE 9 ACTUAL MAIDS BREADBOARD PROTOTYPE.

6.3 MAIDS PCB Board

Before the advent of PCB boards, electronic circuits were constructed using the point-to-point wiring process. Unfortunately, "…this approach led to frequent failures at wire junctions and short circuits when wire insulation began to age and crack." (Sparkfun, 2020) The creation of the PCB board resulted from the electronic industry's move towards the use of integrated circuits (smaller size and lower cost electronic components) and pressure on manufacturers to reduce the size and manufacturing costs.

PCB (an acronym for Printed Circuit Board) is the traditional name for the bare board which supplies the circuit layout and on which the components are mounted. A printed circuit board is used to mechanically support and electrically connect electronic components using conductive pathways, tracks, traces or vias etched from copper sheets laminated onto a non-conductive substrate. A PCB board permits signals and power to be routed between physical devices. The metal material used to make the electrical connections between the surface of the PCB and the electronic components is called solder. Solder, being a metal, serves as a strong mechanical adhesive for the components.

The MAIDS PCB board is composed of alternating layers of different materials which are laminated together resulting in a single object. The base material of the MAIDS PCB board is a solid fiberglass core designated as FR4 which provides the PCB board's rigidity and thickness. The thickness of the PCB board is the standard 1.6 mm (0.063"). A thin layer of copper foil is laminated onto the fiberglass board with heat and adhesive. MAIDS' PCB board contains one ounce of copper per square foot and is double sided board (copper is applied to both sides of the substrate). Each ounce per square foot translates to about 35 micrometers or 1.4 thousandths of an inch of thickness of copper.

The MAIDS PCB board did not have a layer on top of the copper foil called the solder mask layer (usually used to insulate the copper traces from accidental contact with other metal, solder, or conductive bits) nor did it incorporate a silkscreen (adds letters, numbers, and symbols to the PCB).

The final MAIDS PCB board design is shown below.

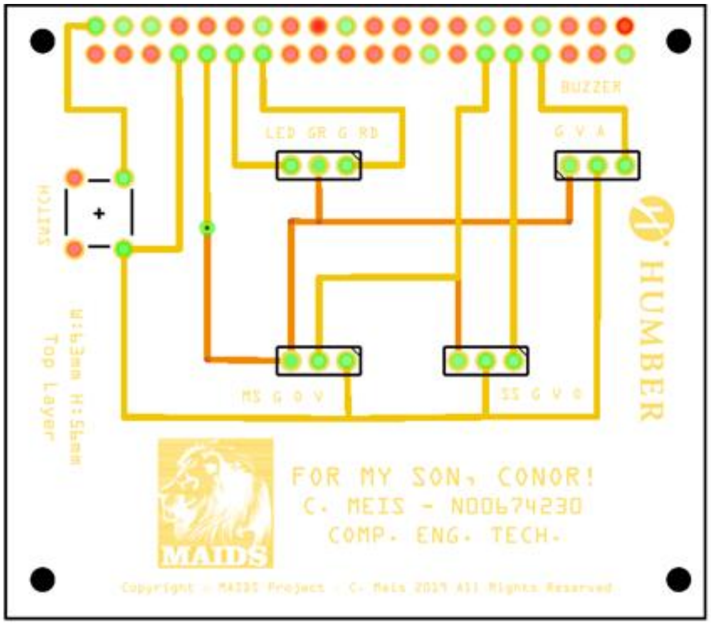


FIGURE 10 MAIDS PCB BOARD DESIGN DIAGRAM.

The PCB board requires the following Raspberry Pi 4 and custom-made PCB board GPIO pin connections:

TABLE 3 RASPBERRY PI 4 AND PCB BOARD CONNECTIONS.

Raspberry Pi and custom-made PCB Board pin connections						
Component	Sensor pin	RP4 pin	Sensor pin	RP4 pin	Sensor pin	RP4 pin
Motion Sensor	VCC	3.3V	Middle pin	Pin 29	GND	GND
LED Module	Green	Pin 9	Red	Pin 11	GND	GND
Sound Sensor	VCC	3.3V	GND	GND	Digital Input DO	Pin 31
Push Button	VCC	3.3V	Signal Pin	Pin 7, Motion Sensor and Sound Sensor		

The final MAIDS PCB board is shown below displaying the top layer with the via (a hole in a PCB board used to pass a signal from one layer to another) connecting the top and bottom layers of the PCB board, the traces for connecting the LED module, sound and motion sensors, and the bottom layer (ground (GND) layer).

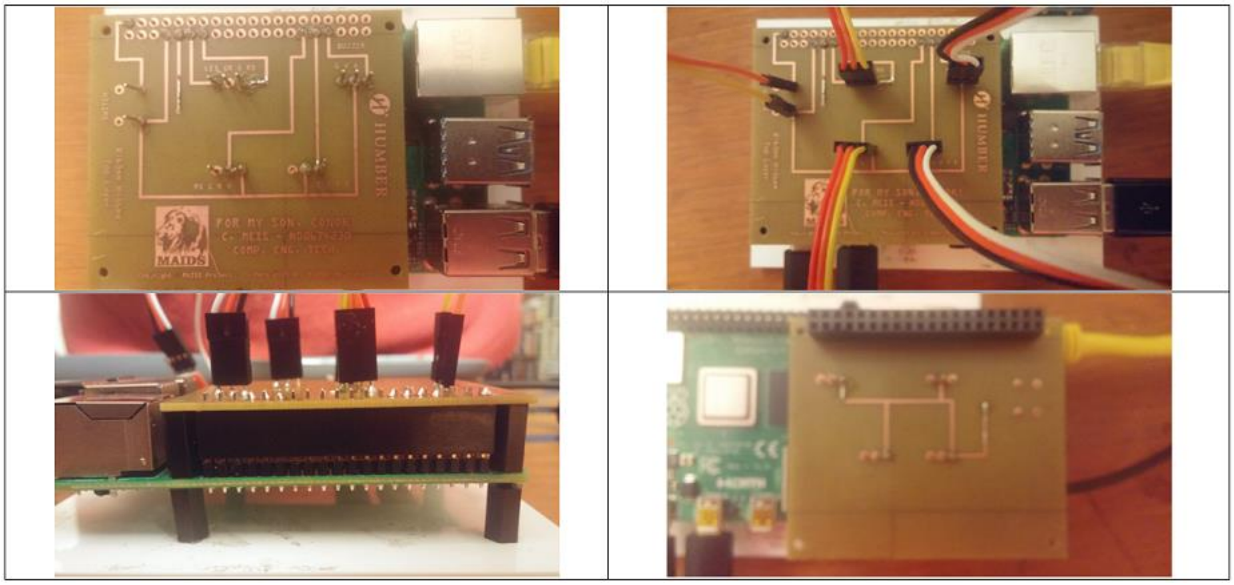


FIGURE 11 PCB BOARD BOTTOM AND TOP LAYERS.

6.4 Bill of Materials

The MAIDS' bill of materials for the design of the PCB board prototype is as follows:

TABLE 4 BILL OF MATERIALS FOR MAIDS PROJECT.

4	MG Chemicals 5" x 3" Copper Clad Board, Double Sided, 1 oz Copper, 1/16" Thick, FR4	587	1	https://www.amazon.ca/MG-Chemicals-Prototyping-1-Ounce-16-Inch/dp/B008OAFOUO/ref=pd_sbs_328_2/132-5081513-4200133?_encoding=UTF8&pd_rd_i=B008OAFOUO&pd_rd_r=95480a50-419b-47d5-91b4-3b88102fc136&pd_rd_w=zfGQl&pd_rd_wg=FhDkE&pf_rd_p=dbebb38c-0e3d-4a67-ac15-432d7c7a2789&pf_rd_r=A5QQ0YC5SPEYDWN36ZA7&psc=1&refRID=A5QQ0YC5SPEYDWN36ZA7
5	Elegoo 120pcs Multicolored Dupont Wire 40pin Male to Female, 40pin Male to Male, 40pin Female to Female Breadboard Jumper Wires Ribbon Cables Kit for Arduino	EL-CP-004	1	https://www.amazon.ca/Elegoo-120pcs-Multicolored-Breadboard-arduino/dp/B01EV70C78/ref=sr_1_1_sspa?keywords=jumper+cables&qid=1579928593&s=industrial&sr=1-1-spons&psc=1&spLa=ZW5jcnlwdGVkUXVhbGlmaWVyPUEyVkZXSjU0U1RGSjFQJmVuY3J5cHRlZElkPUEwMDcxMTkxM0tSVVU5WVBKQV

				hKMiZlbmNyeXB0ZWRBZElkPUEwMzg1ODgyMUwxRTJBRE0zR1dIWCZ3aWRnZXROYW1lPXNwX2F0ZiZhY3Rpb249Y2xpY2tSZWRpcmVjdCZkb05vdExvZ0NsaWNrPXRydWU=
6	AUSTOR Lead Free Solder Wire with Rosin Core 0.6mm	AMA-17-532	1	https://www.amazon.ca/dp/B071XVPJVX/ref=sspa_dk_detail_0?psc=1&pd_rd_i=B071XVPJVX&pd_rd_w=apWS0&pf_rd_p=4b7c8c1c-293f-4b1e-a49a-8787dff31bcb&pd_rd_wg=s5ddm&pf_rd_r=Y6RJTAX1MR0X3E50MH9X&pd_rd_r=b093f5a4-5e04-4c4a-bf07-683669f8db02&spLa=ZW5jcnlwdGVkUXVhbGlmaWVyPUExWUMwVUFXVTBTMzUmZW5jcnlwdGVkSWQ9QTA0Nzk5MzAzSENLOTZDTEhKOTBBJmVuY3J5cHRlZEFkSWQ9QTAyMTk5NjJJTDlaVjVBQUtLRksmd2lkZ2V0TmFtZT1zcF9kZXRhaWwmYWN0aW9uPWNsaWNrUmVkaXJlY3QmZG9Ob3RMb2dDbGljaz10cnVl

6.5 Time commitment

The order lead time, the time from order received to customer order delivered was approximately 2 weeks. The order handling time, the time from customer order received to sales order created was less than 12 hours. The manufacturing lead time, Time from sales order created to production finished (ready for delivery) was approximately 2 weeks. The production Lead Time - Time from start of physical production of first submodule/part to production finished (ready for delivery) was approximately 2 weeks. The Delivery Lead Time - Time from production finished to customer order delivered was approximately 1 week due to testing and slight software modifications. (Rajaniemi, 2012, Measuring and Defining Lead Time in a Telecommunication Production)

The working time, time to design and fabricate the final MAIDS PCB board deliverable, was 1.5 hours per work day, five days per week (total of 7.5 hours per week), for a total of 15 hours, total.

6.6 Testing

The testing phase of the PCB board included the following:

- Applying the DRC rules in Fritzing to identify possible trace overlaps that might lead to short circuits which were corrected as before passing on to the manufacturing process.
- Once manufactured, the board was meticulously inspected visually for board discolorations, broken traces, correct connections and cracks on the board itself. By examining the board and the surface-mounted components, one can identify obviously damaged or disconnected parts before beginning testing. In addition, the boards was inspected for obvious signs of oxidation and corrosion such as rust. Wires were also inspected to make sure all of the components were connected. PCB boards was found to be physically sound and correctly connected from node to node.
- A magnifying glass was used to identify tin whiskers between pads and solder joints along with tin bridges. None was found.
- No cracks or blobs of solder were found.
- Opens tests were performed with a multimeter to make sure currents flowed between nodes by means of a LED-Resistor component placed between connecting points (nodes) to check for electrical conductivity. All traces and vias were found to carry currents properly.
- Shorts-test were performed using a multimeter were the resistance between neighboring traces and pads were measured on a PCB resulting in high resistance. No shorts.

6.7 PCB Board Final Status Report

Prepared by Claudio F. Meis, February 12, 2020.

The presentation of the MAIDS project at the Capstone Project EXPO at 1:00 – 4:00 p.m. on Thursday, April 9, 2020, is still on track.

The following work has been completed on the MAIDS PCB board:

- PCB Board schematic design.
- PCB Breadboard design (Fritzing and actual)
- PCB board fabrication (laser etching)
- PCB board testing.

Progress against Milestones

Schematic Design

Milestone 100%

Progress 100%

Breadboard Design

Milestone 100%

Progress 100%

PCB Board Fabrication

Milestone 100%

Progress 100%

PCB Board Testing

Milestone 100%

Progress 100%

Key Issues

No issues need to be resolved to meet Capstone Project EXPO deadline.

Action Steps

None.

7.0 Printed Circuit Board

The success of any project is often dependent on the foundations it is built upon. Much in the same way, the success of any electronic device depends on what it is built on. The PCB board of any electronics device relays electrical signal that performs some function for the equipment. Be it the communication signal between Raspberry Pi 4 and the custom-made PCB board, or a simple on-off signal from the switch, the effectiveness of the design is a function of the capabilities offered by the PCB board itself. A Printed Circuit Board (PCB) does not just connect electrical components using etched copper pathways, but also provides mechanical strength to it.

This section of the report does not concentrate on the actual fabrication process but on the specification, guidelines, considerations and recommendations required to produce an error-free PCB board. The actual fabrication process is delineated in sections 3.1.2.1 PCB Board Manufacturing Process and 3.1.3.2 PCB Board Cutting and Etching of this report.

7.1 PCB board Design Flow

In order to design a successful MAIDS' PCB board careful thought was given to its design flow. The design flow for the MAIDS project consisted of six major procedures:

1. Logic design((section 3.2.11.1 MAIDS Schematics)
2. Design verification by circuit simulation (section 3.2.11.1 MAIDS Schematics)
3. Schematic design (section 3.2.11.1 MAIDS Schematics)
4. PCB design (section 3.2.11.1 MAIDS Schematics)
5. Fabrication Specifications of the PCB board (This section)
6. Testing of PCB board (section 3.2.11.1 MAIDS Schematics)

Each section stated above will provide information on the particular part of the design flow process.

7.2 PCB Fabrication Specifications

PCB board manufacturing begins with the user-generated artwork that is then sent to the manufacturing facility in a particular format (RS-274X Gerber file) to be a laser etched. MAIDS used the following three standard technologies during the manufacturing of the PCB board: Machining, Imaging and Etching.

7.2.1 Machining

Machining includes drilling, punching holes and routing on a PCB with laser cutting. The strength of the board needs to be taken into account while machining hole-diameters accurately. Small holes were avoided so that plating was easily accomplished.

7.2.2 Imaging

Imaging transfers the circuit artwork onto individual layers. MAIDS' double sided PCB board design used direct laser imaging for creating the patterns on a print-and-etch basis.

7.2.3 Etching

Etching refers to the removal of unwanted metal and dielectric from the board that takes place by either dry or wet processes. MAIDS used a dry process. The uniformity of etching is the prime concern in this stage.

7.3 PCB Board Specifications

The specifications used in the design and fabrication of the MAIDS PCB board are listed in the table below.

TABLE 5 PCB BOARDS DESIGN SPECIFICATIONS.

Parameter	Standard
Annular ring: Internal Minimum Pad Size	.014" larger than finished hole size
Annular ring: External Minimum Pad Size	.014" larger than finished hole size
Plane Layer Clearance	
Plane Layer Clearance - PTH & NPTH Hole to Inner Layer Trace	.015" Spacing
Inner Layer Clearances:	0.010”
Copper to Edge of PCB:	0.010” for outer layers, 0.015” for inner layers, 0.020” is preferred.
Pad Size/Annular Ring:	Pad size should be +0.010” over the finished hole size for Vias +0.014” over the finished hole size for Component holes
Hole Size:	0,008” minimum finished hole size, 0.015” or larger hole size recommended
Copper Trace Width/Spacing:	0.005”) trace widths Recommended minimum spacing: 8 mils.
Inner Layer Line width on 1 ounce copper	Standard-.005"

7.3.1 PCB Board Component Placement

The component placement stage of the PCB layout process is very important. How the designer of the PCB board places the electronic components determine how easy the board is to manufacture, as well as how well it meets the original PCB design requirements. The MAIDS project used the following general board layout guidelines to place the components on the PCB board:

1. Orientation: All similar components were placed in the same direction. This helps the operative routing of the PCB board design, as well as, to help ensure a well-organized soldering process during assembly.
2. Placement: Placing components on the solder side of a board that would rest behind plated through-hole components should be avoided.
3. Organization: All through-hole (TH) components should be placed on the top side of the PCB board to minimize the number of assembly steps.

7.3.2 PCB Board Copper Thickness

Copper thickness of PCB boards can be specified directly or as the weight of copper per area (in ounce per square foot). One ounce per square foot is 1.344 mils or 34 micrometers thickness. MAIDS uses the common FR-4 substrate with one ounce copper per ft^2 (35 µm) which is the most common thickness;

7.3.4 PCB Board RoHS Compliance

Manufacturers, retailers and suppliers of electrical and electronic products in Canada need to comply with regulations stipulated in the Restriction of Hazardous Substances directive (RoHS2 Directive, 2011/65/EU). The directive bans the use of lead (among other heavy metals) in consumer items. MAIDS’ PCB board is RoHS-compliant, meaning that all manufacturing processes did not involve the use of lead, all solder used was lead-free, and all components mounted on the board were free of lead, mercury, cadmium, and other heavy metals.

7.3.5 PCB Board Laminate

FR-4 is by far the most common material used today. The board stock with un-etched copper on it is called: copper-clad laminate. FR-4, is a woven fiberglass cloth impregnated with an epoxy resin (also known as polyepoxides, are a class of reactive polymers which contain epoxide groups).It provides low water absorption (up to about 0.15%), good insulation properties and good arc resistance. Several grades with somewhat different properties are available and is typically rated to 130 °C. The MAIDS’ PCB board uses an FR-4 laminate in its fabrication.

7.3.6 PCB Board Trough-Hole Technology

Through-hole technology (also spelled "thru-hole"), refers to the mounting scheme used for electronic components that involves the use of leads (wire or a metal pad designed to connect two locations electrically) on the components that are inserted into holes drilled in printed circuit boards (PCB) and soldered to pads on the opposite side by manual assembly (hand placement). MAIDS’ double-sided PCB board used through-hole technology. It is worth noting that Through-hole manufacturing adds to board cost by requiring many holes to be drilled accurately. Through-hole technology used holes and vias with a diameter of 0.008”.

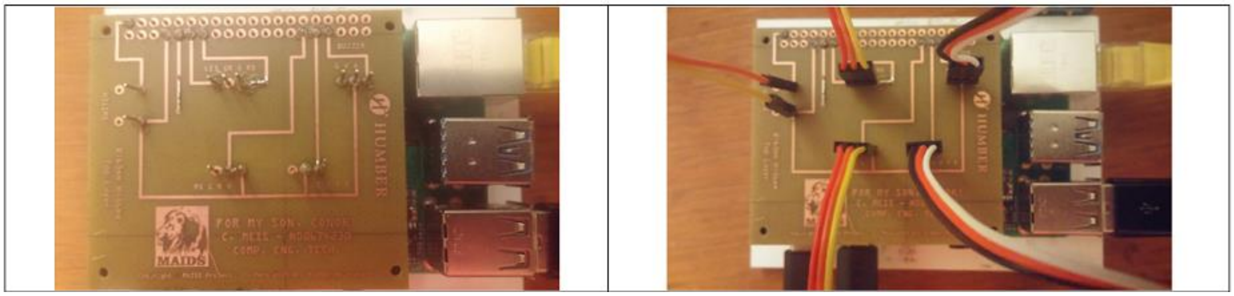


Figure 10: Final MAIDS PCB board.

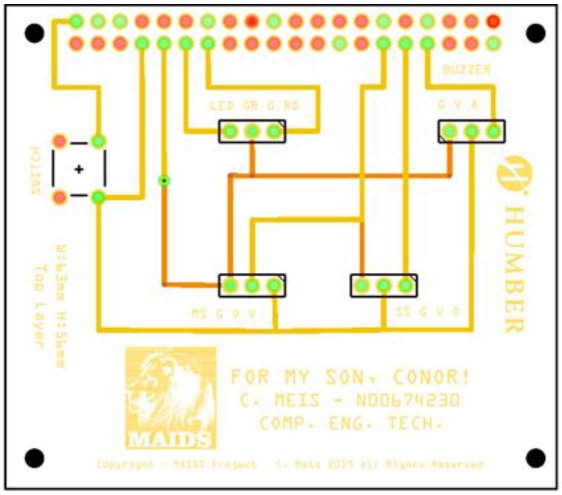


Figure 11: Final PCB Board Design Using Fritzing Software.

7.3.7 Routing Guidelines for PCB Layouts

MAIDS' PCB layout followed the recommended best practices to achieve a trouble-free layout. In the MAIDS project, traces were placed as directly as possible between components, as well as, providing the shortest path between them. Is it also worth noting that if component placement forces horizontal trace routing on one side of the board, then the designer should route traces vertically on the opposite side.

7.3.8 PCB Board Size and Shape

MAIDS’ PCB board was designed to be rectangular in shape and as small as possible. The board’s size of 63 mm x 56 mm assured that the PCB board did not overlap the Raspberry PI 4 and therefore needlessly increase the size of the projects enclosure. Also, the chosen shape and size design made sure that larger Through-hole components had enough space to make them easier to solder onto the board.

7.3.9 PCB Board Trace Angles and Widths

Professionally designed PCB boards, have most of the copper traces bend at 45° angles. One reason for this is that 45° angles shorten the electrical path between components compared to 90° angles. Another reason is that high speed logic signals can get reflected off the back of the angle, causing interference. Unfortunately, CENG317 course directives stated that traces between components should bend at 90° angles. This directive should be changed in the future to mitigate the problems stated above and produce professional PCB board designs.

Like layer thickness, the width of the PCB board traces affects how much current can flow through the circuit without damaging the circuit. The proximity of traces to components and adjacent traces will also determine how wide your traces can be. MAIDS’ small PCB board design did not have many traces and components. The MAIDS trace width was calculated (0.51 mil) using the PCB Trace Calculator (Bittele Electronics, 2020) and employing the following parameters:

TABLE 6 TRACE CALCULATION PARAMETERS.

Field	Value	Units
Current (max. 35A)	50	mA
Copper Thickness	1	oz/ft2
Temperature Rise (max. 100°C)	10	°C
Ambient Temperature	25	°C
Conductor Length	1	inch
Peak Voltage	3.3	Volts
Trace width required	0.51	mil

However, having designed the PCB board in Fritzing software, the minimum trace width was set to 8 mils.

7.3.10 PCB Board Verification

MAIDS' verification process included Fritzing's Design Rule Check (DRC). DRC imposes limitations on the PCB board layout in order to ensure its successful manufacturing. The common design rules applied to MAIDS were: minimum trace spacing, minimum trace width, minimum drill diameter, and trace overlapping.

7.3.11 PCB Board Final Status Report

Prepared by Claudio F. Meis, February 15, 2020.

The presentation of the MAIDS project at the Capstone Project EXPO at 1:00 – 4:00 p.m. on Thursday, April 9, 2020, is still on track.

The following work has been completed on the MAIDS PCB board:

- PCB Board design flow.
- PCB Board Fabrication Specifications
- PCB Board Specifications
- PCB Board testing.

Progress against Milestones

PCB Board Design Flow

Milestone 100%

Progress 100%

PCB Board Fabrication Specifications

Milestone 100%

Progress 100%

PCB Board Specifications

Milestone 100%

Progress 100%

PCB Board Testing

Milestone 100%

Progress 100%

Key Issues

No issues need to be resolved to meet Capstone Project EXPO deadline.

Action Steps

None.

8.0 Enclosure

8.1 Enclosure Design Software

The MAIDS’ enclosure was designed using CorelDraw 2018 for precise laser cutting running under the Windows 10 operating system. CorelDraw 2018 can export either .svg or .pdf files which are the preferred files for laser cutting.

8.2 Enclosure Design Software

The MAIDS’ design document size was restricted to be within the maximum laser table size of 12″ high by 24″ wide. The document was oriented in landscape and ensuring that any artwork is constrained within the 12″x 24″ document size.

8.3 Enclosure Design Stroke Width and Line Colors

The laser cutter uses vector lines to cut with the Stroke Width set to Hairline (0.000 mm wide). Outside laser cuts are colored green while inside cuts are colored red. Etching of words or logos requires any other color and lines thicker than 0.000 mm (hairline) which will result in the burning a light layer off of the top of the material.

8.4 Enclosure Design and Materials

The enclosure for the MAIDS project used 3 mm thick white acrylic sheet as its base material. The design incorporated a small footprint, hollow-shell, 10 layer-stacked model designed for easy assembly and in order to reduce desktop footprint, provide boards protection as well as weight reduction.

8.5 Enclosure Design Heat Dissipation Consideration

Design of the MAIDS' enclosure implements heat dissipation measures. The case includes the following heat dissipation components:

1. Four aluminum heat sinks (CPU, memory, Ethernet and USB). The properties that make aluminum heatsinks appropriate for the MAIDS project are: Good thermal and electrical conductivity, Low density with a density ~ 2,700 kg/m3, Low weight, High strength of between 70 and 700 MPa, Easy malleability, Excellent corrosion resistance, non-magnetic which avoids interference of magnetic fields and Easy to recycle. (Radian, 2020)

2. A 30 mm x 30 mm fan for heat dissipation (connected to 3.3 V DC and GND GPIO pins on board).

8.6 Enclosure Design Physical Characteristics

The MAIDS' enclosure physical characteristics are as follows:

1. 85 mm (length) x 56 mm (width).

2. Accommodates holes for:

    a. A C-Type power connector

    b. Two micro HDMI connectors

    c. An audio port

    d. Port for USB 3.0

    e. Port for USB 2.0

f. Port for Ethernet connectors.

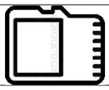
g. Port for a display device

h. Port for a camera connector.

8.7 Enclosure Design Etched Icons

The MAIDS' enclosure integrates case icons and logo clearly identifying available connectors. The icons and logo are etched onto the outward-facing enclosure surfaces and are as follows:

TABLE 7 MAIDS ICONS AND LOGOS.

Icon Description	Icon Image
I Love You Son	
HDMI	
USB2 and USB3	
SD Card	
Sound	
Power Supply	
Internet	
MAIDS Logo	

8.8 Enclosure Final Deliverable

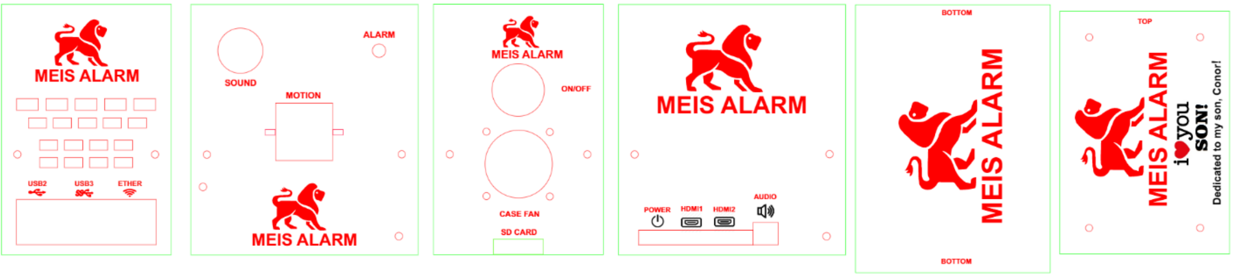
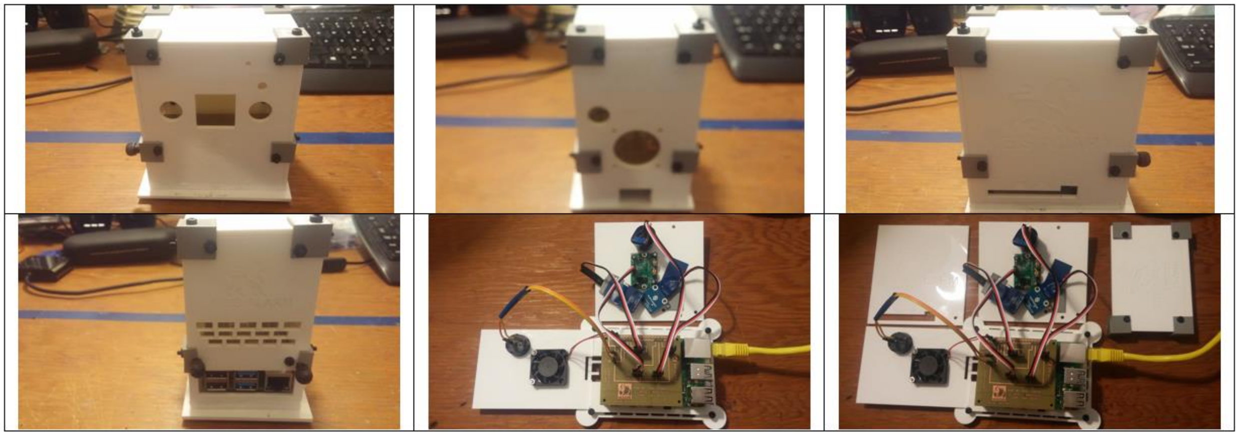
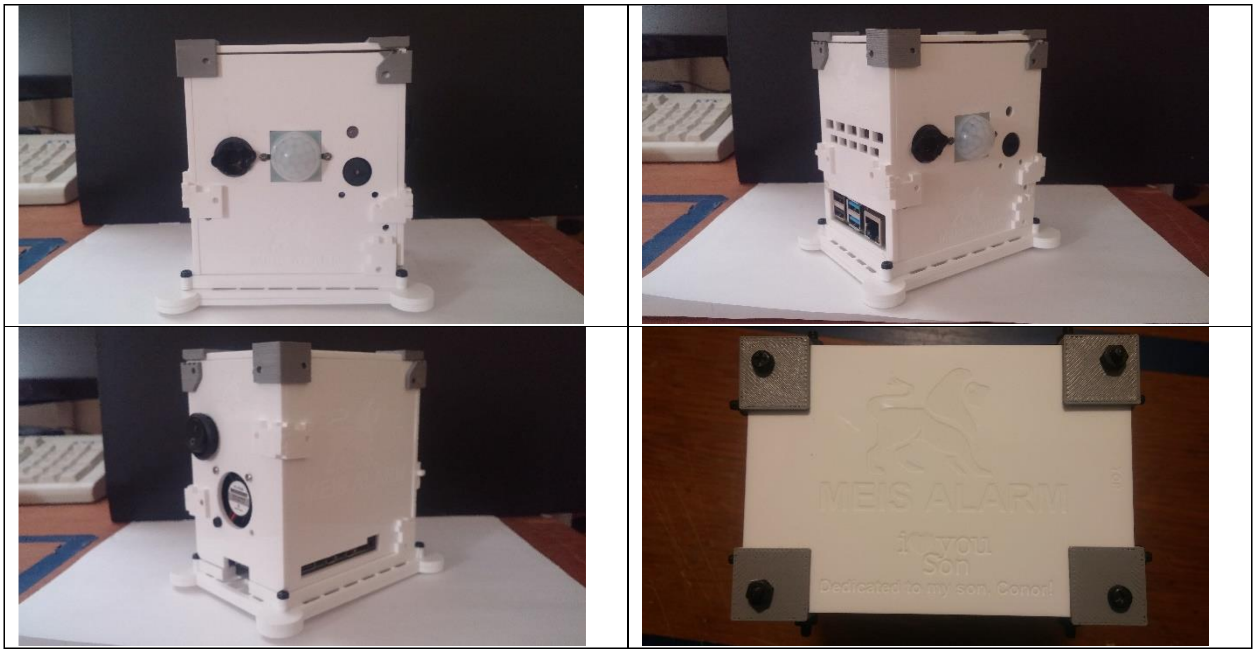


FIGURE 12 MAIDS ENCLOSURE ARTWORK DESIGN.





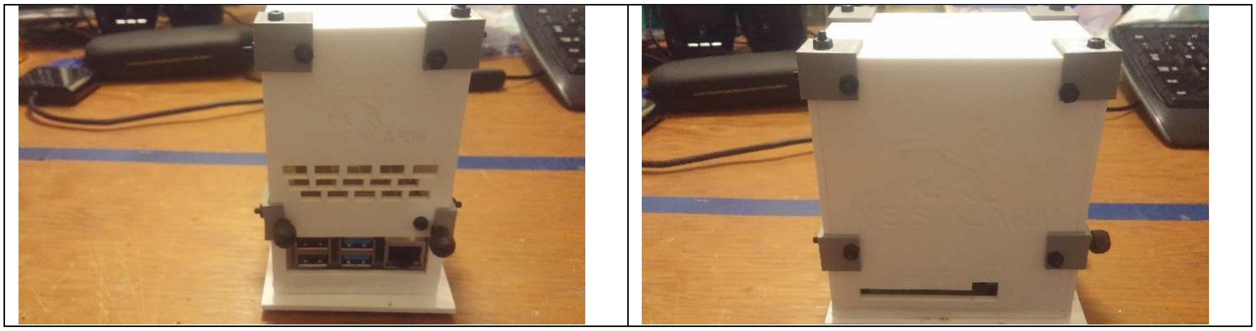


FIGURE 13 MAIDS FINAL ENCLOSURE DELIVERABLE.

8.9 Enclosure Final Status Report

Prepared by Claudio F. Meis, February 15, 2020.

The presentation of the MAIDS project at the Capstone Project EXPO at 1:00 – 4:00 p.m. on Thursday, April 9, 2020, is still on track.

The following work has been completed on the MAIDS PCB board:

- Enclosure design.
- Enclosure Specifications.
- Enclosure Fabrication.
- Enclosure Deliverable.

Progress against Milestones

Enclosure Design

Milestone 100%

Progress 100%

Enclosure Specifications

Milestone 100%

Progress 100%

Enclosure Fabrication

Milestone 100%

Progress 100%

Enclosure Deliverable

Milestone 100%

Progress 100%

Key Issues

No issues need to be resolved to meet Capstone Project EXPO deadline.

Action Steps

None.

9.0 Integration

Electronics integration is the art of merging audio, video, and control systems into one seamless network of interrelated devices. MAIDS' technologies allow for dramatic possibilities with ease-of-use interfaces, bringing different platforms under the user's control, all with a single display and control interface. MAIDS provides easy-to-use integration solution allowing it to optimize the functionality and impact of the systems and providing the best outcome and value.

The MAIDS' system integrates all of the hardware (electronic components) and software (python code, Android application and web services) into a functional system that is easy for anyone to use. On the one hand, the integration of hardware components include:

a. LED sensor
b. Motion sensor
c. Sound sensors
d. USB camera
e. Custom PCB board
f. Raspberry Pi 4 platform.

On the other hand, the software components include:

a. Android phone application
b. Web services (Twilio, email and PushNote).

The system was customized to work with the required electronics (Raspberry Pi 4 embedded system and custom-made PCB board) and software, and designed from scratch. In addition, the MAIDS Home and Remote integration system gives the user the ability to control the system remotely from their smart phone or tablet.

9.1 Data Sent by Hardware: Motion/ Sound Sensors and Processor

A sensor is a device which produces an output by detecting the changes in quantities or events. Generally, sensors produce an electrical signal or optical output signal due to a physical change in some characteristic that changes in response to some excitation. Digital Sensors produce a discrete digital output signal or voltage that are a digital representation of the quantity being measured. Digital sensors produce a binary output signal in the form of a logic “1” or a logic “0”, (“ON” or “OFF”) and are typically linked to a control program that specifies acceptable levels. This means then that a digital signal only produces discrete (non-continuous) values which may be outputted as a single “bit”, (serial transmission).

There are two different types of digital sensors in the MAIDS device: a sound sensor and a motion sensor. Both produce a corresponding digital signal due to changes in quantities or events. The control program decides what to do next based on the data it is fed by the sensors.

On the one hand, the sound sensor produces a HIGH (1) output through its digital output pin when a change is sound levels is detected. The digital signal is then carried by the connecting wire to the input pin of the custom-made PCB board which transfers the signal to the GPIO pin 11 of the Raspberry Pi 4 for processing.

On the other hand, the motion sensor produces a HIGH (1) output through its digital output pin when a change is radiation levels are detected. The digital signal is then carried by the connecting wire to the input pin of the custom-made PCB board which transfers the signal to the GPIO pin 13 of the Raspberry Pi 4 for processing.

For its part, the Central Processing Unit (CPU) is the part of a computer system that is commonly referred to as the "brains" of a computer. The CPU is responsible for executing a sequence of stored instructions (program). This program will take inputs from an input device (motion/sound sensors), process the input in some way and output the results to an output device (i.e. LED module, web services, etc.).

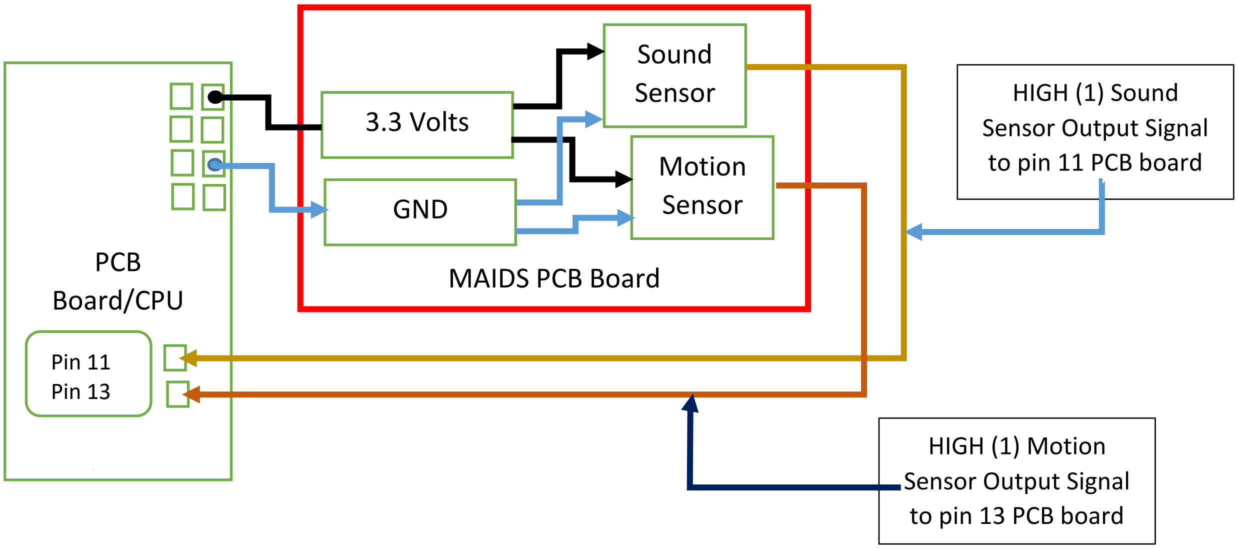

Sound Sensor
3.3 Volts
Motion Sensor
GND
MAIDS PCB Board
PCB Board/CPU
Pin 11
Pin 13
HIGH (1) Sound Sensor Output Signal to pin 11 PCB board
HIGH (1) Motion Sensor Output Signal to pin 13 PCB board

FIGURE 14 DATA INPUTS FROM MOTION/SOUND SENSORS TO RASPBERRY PI 4.

9.2 Data Retrieved By Mobile Application

The Android application is composed of two activities: login activity and Data/Control activity. One of the main functions of the data and control activity is to retrieve data from the remotely located MAIDS device. In particular, the Android application will retrieve intrusion data (i.e. time of intrusion, intrusion location, contact information, etc.) from the MySQL database located on the MAIDS device (database server) and real-time statistical information. By clicking on the “Retrieve Database Info” button, the Android application, through the wireless Internet, connects to the MAIDS device, locates and extracts from the database the information required, and then, transmits the information back to the Android application and displays it on the screen via a WebView element. Similarly, clicking on the “Retrieve RASPI Stats”, the Android application connects to the MAIDS device, retrieves real-time statistical information about the Raspberry Pi platform and then transmits the information back to the Android phone or tablet to be displayed on the screen via a WebView element.

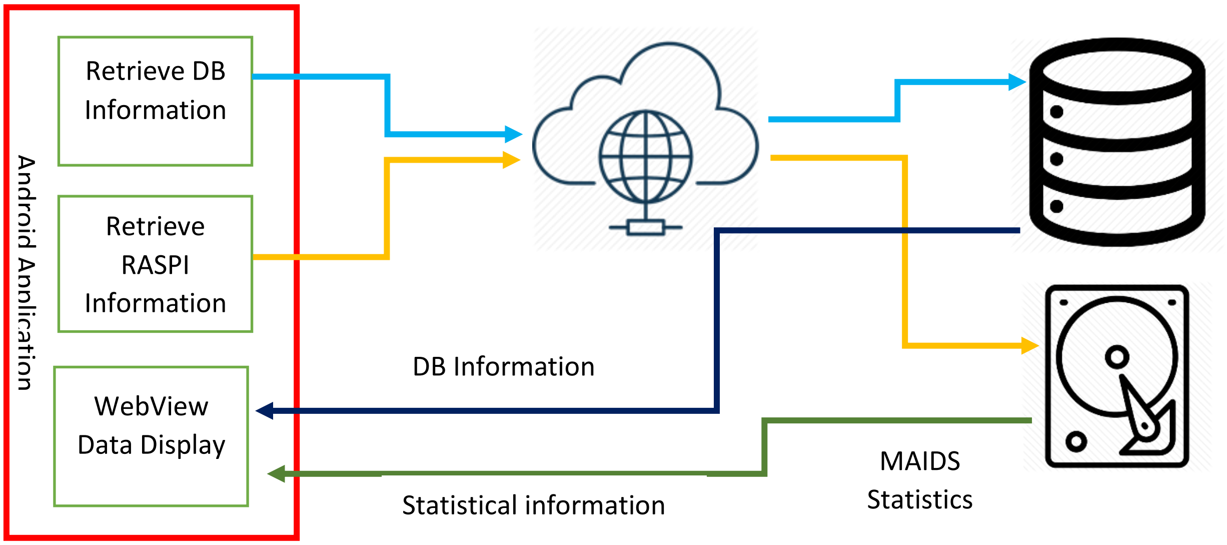


FIGURE 15 DATA RETRIEVED BY MOBILE APPLICATION.

9.3 Action Initiated By Mobile Application

The second Android application activity is the Control activity. The three main functions initiated by the mobile application through the control activity are as follows:

1. Activate MAIDS device (Turn it ON)
2. Deactivate MAIDS device (Turn it OFF)
3. Test LED module

By clicking the “Activate MAIDS” button, the Android application, through the wireless Internet, connects to the MAIDS device, and then executes the Python3 program named maids_final_python_code_22102019_backup1.py. The program then proceeds to execute the code that initiates the monitoring capabilities of the MAIDS device.

Similarly, by clicking the “Deactivate MAIDS” button, the Android application, through the wireless Internet, connects to the MAIDS device, and then executes the Python3 program named MAIDSOFF.py. The program then proceeds to execute the code that terminates the monitoring capabilities of the MAIDS device.

In addition, by clicking the “Test LED Module” button, the Android application, through the wireless Internet, connects to the MAIDS device, and then executes the Python3 program named TestLEDS.py. The program then proceeds to execute the code that sequentially turns the green and red LEDs thereby testing the functionality of the LED module of the MAIDS device.

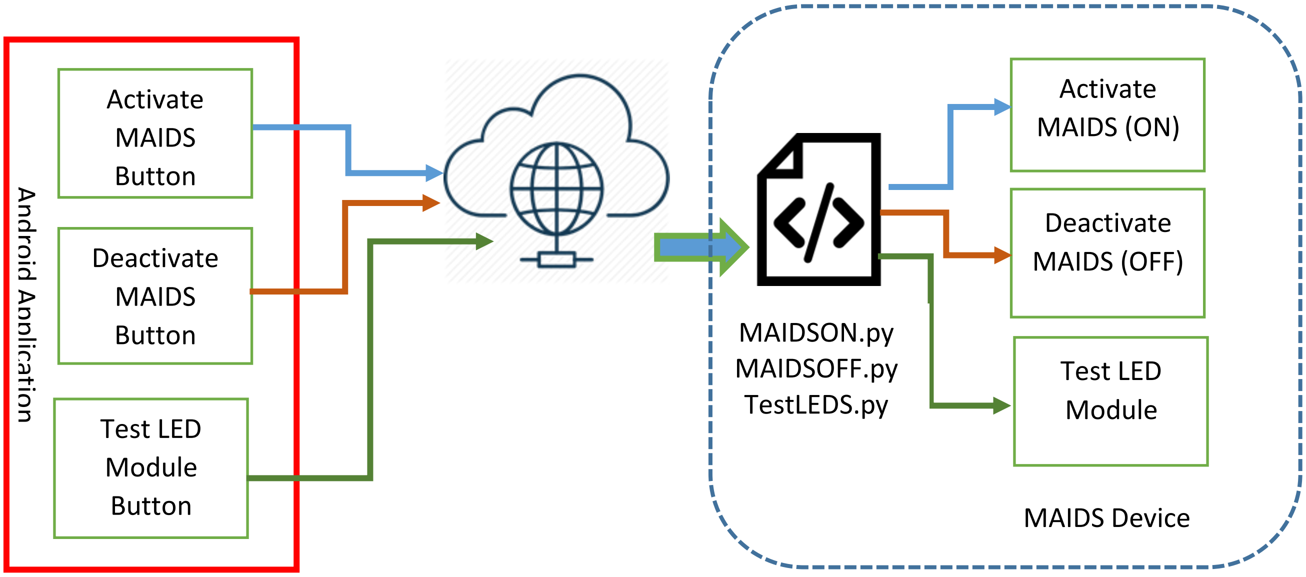


FIGURE 16 ACTIONS INITIATED BY MOBILE APPLICATION.

9.4 Action Received By Hardware

From the data and control activity there are several actions received by the MAIDS device. These are:

1. **LED Module Functionality Test:** The test is carried out when the MAIDS device receives via the internet the command to execute the TestLEDS.py program located in the /home/cmeis/maids1 directory.
2. **LED Module Visual Display of Intrusion:** If motion or sound is detected by the motion (PIR) or sound sensors on the MAIDS PCB board, they will generate an input signal. The motion/sound input signal travels through the connecting wires to the GPIO pins to the Raspberry Pi 4 Model B and the firmware processes the input signal on the particular input GPIO pin and executes the following functions (intruderwarning(), notify() and siren()) which in turn play an audible warning of the intrusion, runs the notification protocol (email with picture, push notification, phone call and SMS message), and finally plays a loud siren sound. The code for the actions mentioned is found in the function code of the maids_final_python_code_22102019_backup1.py program.
3. **Activation of MAIDS device:** The MAIDS device executes the commands found in the maids function maids_final_python_code_22102019_backup1.py program. Its sole purpose is to run the executable code that sets up and initiates the monitoring capabilities of the MAIDS device.
4. **Deactivation of MAIDS device:** The MAIDS device executes the commands found in the program MAIDSOFF.py program found in the /home/cmeis/maids1

directory. Its sole purpose is to run its executable code which deactivates the MAIDS device.

5. **Snap Intrusion Photo by Webcam:** If motion or sound is detected by the motion (PIR) or sound sensors on the MAIDS PCB board, they will generate an input signal. The motion/sound input signal travels through the connecting wires to the GPIO pins to the Raspberry Pi 4 Model B and the firmware processes the input signal on the particular input GPIO pin and executes send_mail() function. The function executes the fswebcam application and sets the intrusion picture parameters as follows: resolution at 1280x720 pixels, a picture title ('MAIDS INTRUSION ALERT'), an intrusion subtitle ('1234 BROOK ROAD, ETOBICOKE, ON.'), a timestamp for the intrusion ('%Y-%m-%d %H:%M (%Z)'), information on the place of intrusion (‘LIVING ROOM ENTRY'), and the directory where the intrusion picture is to be saved along with its name (“/home/pi/webcam/image.jpg").
6. **Add/Retrieve data to/from MAIDS Database:** When the motion and sound sensors are triggered a signal is produced that executes the appendtodb() function of the maids_final_python_code_22102019_backup1.py program. The device then executes its code whose sole purpose is to add the following information to the MySQL database on the MAIDS device:
    a. Intrusion address
    b. Intrusion location (Room)
    c. Intrusion date
    d. Reporting Person

e. Contact Phone Number

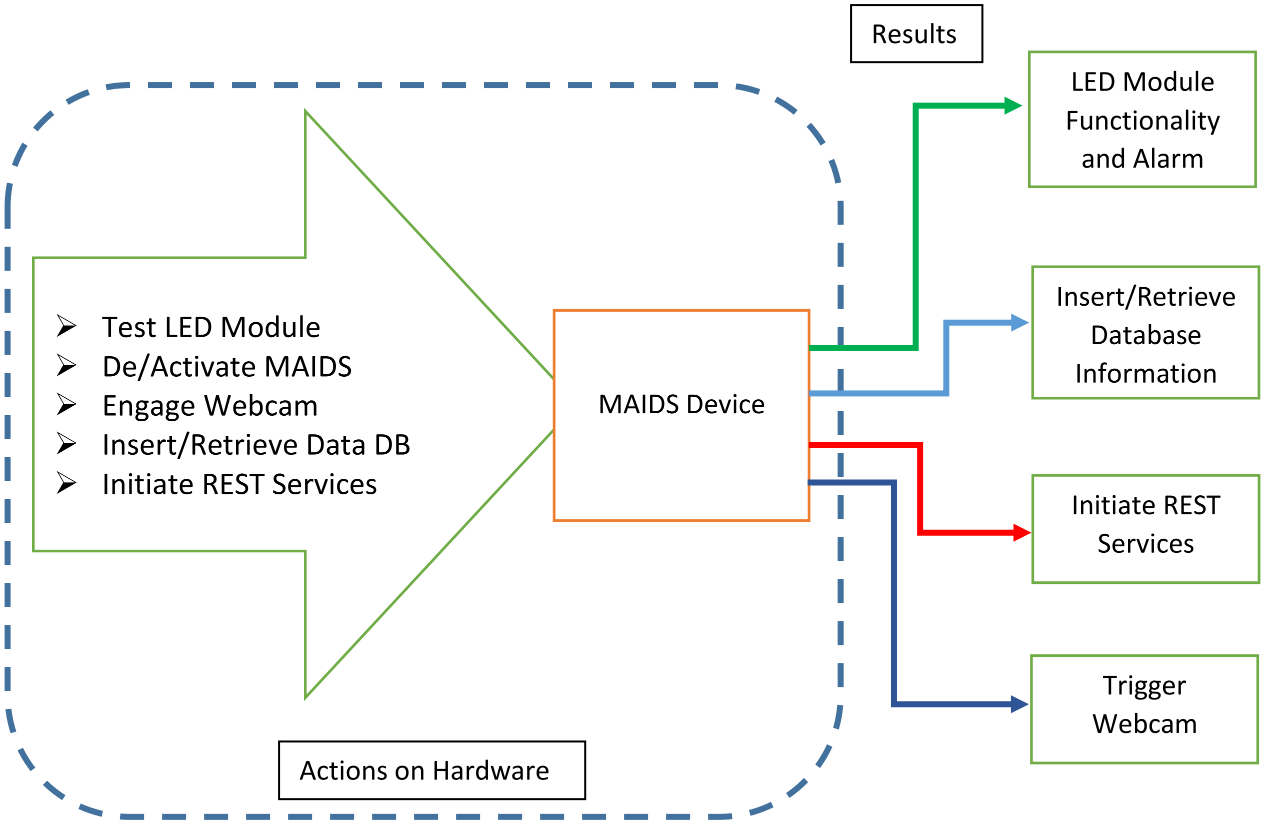
f. Contact Email

7. **Initiate REST Services:** The hardware initiates the Representational State Transfer (REST) services which relies on a stateless, client-server, cacheable communications protocol when detecting an intrusion. Upon triggering of the sound or motion sensors a signal is produced that engages the following three functions in the maids_final_python_code_22102019_backup1.py program:

   a. send_androidpush(): The function calls executes the code for the pushover service using a token ID and user ID. The push notification produces includes information about the intrusion and is sent to the telephone number provided when registering to the service.

   b. sendsms(): Using the Twilio rest service with its own account ID and authorization token, the service sends an SMS message to the telephone number provided when registering to the service.

   c. callphone(): Using the Twilio rest service with its own account ID and authorization token, the service calls the telephone number provided when registering to the service and relays an intrusion message to the user.



- Test LED Module
- De/Activate MAIDS
- Engage Webcam
- Insert/Retrieve Data DB
- Initiate REST Services

FIGURE 17 ACTIONS RECEIVED BY MAIDS HARDWARE.

9.5 Integration Final Status Report

Prepared by Claudio F. Meis, March 2, 2020.

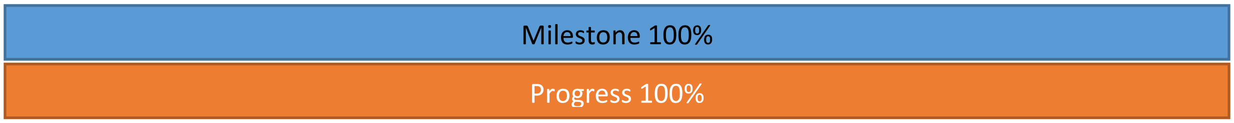
The presentation of the MAIDS project at the Capstone Project EXPO at 1:00 – 4:00 p.m. on Thursday, April 9, 2020, is still on track.

The following work has been completed on the MAIDS PCB board:

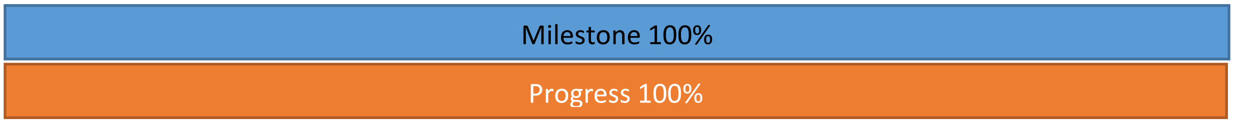
- Hardware Integration.
- Software Integration.

Progress against Milestones

Hardware Integration



Software Integration



Key Issues

No issues need to be resolved to meet Capstone Project EXPO deadline.

Action Steps

None.

10.0 Enterprise Wireless Connectivity

10.1 Enterprise Wireless Connectivity

All businesses have the increasing need to change their enterprise networks in order to accommodate the rapidly growing demand for wireless connectivity. Many enterprises, such as Humber College, are now compelled to build their own enterprise wireless network to support the influx of mobile devices, such as, Internet of Things, and cloud-based applications, as well as increased adoption of WiFi-hungry practices like Bring Your Own Device (BYOD). An enterprise-grade wireless network is more than just a collection of WiFi Access Points (APs). At minimum, it's characterized by superior security and performance; centralized configuration and management; and a much higher capacity for user density. A wireless network is very similar to a wired network with one big difference: Devices don't use cables to connect to the router and one another. Instead, they use radio wireless connections called Wi-Fi (Wireless Fidelity), which is the 802.11 networking standards supported by the Institute of Electrical and Electronics Engineers (IEEE).

In regards to the MAIDS project, wireless connectivity was achieved using the Humber College wireless network with the following set parameters:

a. Range: Up to 200 m
b. Frequency: 2.4 GHz 5 GHz
c. PHY Throughput: Up to 72 Mbps
d. Network type Peer-to-peer: Star

The Humber College wireless network security is implemented using Wi-Fi Protected Access (WPA2) encryption and multiple layer of security like TLS. In addition, Network address translation (NAT) is a method of remapping one IP address space into another. With NAT, one public IP address can hide a number of private IP addresses. Finally it employs Quality of Service (QoS). QoS provides the ability to prioritize different applications, users, or data flows to guarantee a certain level of performance. The actual code used in the wpa_supplicant file to connect to the Humber College wireless network is found in Appendix II of this report.

10.2 MAIDS Database Accessibility: Prototype and Mobile Application

10.2.1 LAMP Installation and Configuration

In order to access the MAIDS database, it must first be installed and configured. MySQL installation and configuration requires a series of prior steps to install it using the LAMP stack. MySQL is a popular relational database system (RDBS). A relational database stores data in a structured format, using rows and columns (Christensen, 2017) and is commonly included in the LAMP (an acronym of the names of the original four open-source components: Linux, Apache, MYSQL, and PHP) stack. MYSQL is a key component of dynamic websites and the best way to store data for web applications. MySQL allows the user to store and maintain large amounts of data easily.

10.2.2 Minimum MySQL Hardware Requirements

The minimum hardware requirement to run the MAIDS' MySQL database include the following:

a. CPU: Intel Core or Xeon 3GHz (or Dual Core 2GHz) or equal AMD CPU
b. Cores: Single (Dual/Quad Core is recommended)
c. RAM: 4 GB (6 GB recommended)
d. Graphic Accelerators: nVidia or ATI with support of OpenGL 1.5 or higher
e. Display Resolution: 1280 × 1024 is recommended, 1024×768 is minimum.

10.2.3 LAMP Stack Installation

Installing the LAMP stack requires the procedure outlined in the following sections.

10.2.3.1 Update the Raspbian Operating System

1. Update the Raspbian OS executing the following commands: sudo apt-get update and sudo apt-get upgrade –y.

10.2.3.2 Install Apache2 Server

To install Apache2 Server execute the following steps and command sequence:

1. Install the Apache2 server and restart it executing the following steps and commands: sudo apt-get install apache2 –y and sudo systemctl restart apache2.
2. Once apache2 is installed execute ifconfig eth0 command to display the IP address of the Raspberry Pi.
3. Then, enter the IP address into the browser's address bar and if installed properly, the user should be presented with the Apache2 Debian Default Page.

10.2.3.3 Install PHP Package

To install PHP execute the following steps and command sequence:

1. Install PHP package and dependencies
   a. sudo apt-get install php libapache2-mod-php –y
2. Create an index.php file with the following content: <?php echo "hello world"; ?>.
3. Using the web browser, enter into the browser's bar the following: http://xxx.xxx.xxx.xxx/index.php
4. A correctly installed PHP package will display the following message: "Hello World".

10.2.3.4 Install MySQL Database (mariadb)

To install the MySQL database execute the following steps and command sequence:

1. Install the maridb database and dependencies: sudo apt-get install mariadb-server mariadb-client php-mysql –y
2. Then, restart the Apache2 server executing the command: sudo systemctl restart apache2
3. Finally, secure the MySQL installation with the command: mysql_secure_installation

At this point, the LAMP installation is finished.

10.3 MySQL Database Password Configuration Procedure

Once installed, MySQL has no root password and it won't work with MariaDB unless one is specified. Configuring MySQL and assigning a password to the root user requires the following procedure:

1. Start the MySQL database: sudo mysql -u root
2. Enter root user password (specified during MySQL installation)
3. reset the root password with the following statement:
    a. ALTER USER 'root'@'localhost' IDENTIFIED BY 'xxxxxxxxxxxxxxxx';
4. Quit MySQL and login again to create the MAIDS database.
5. Follow steps 1 and 2 (with new root password)

10.4 Create maids1 Database and maidsintrusion Table

1. Execute the following command to create a database:
   a. CREATE DATABASE maids1;
2. Choose the maids 1 database executing the command:
   a. use maids1;
3. Next, create the table ‘maidsintrusion’ with the CREATE TABLE statement as follows:

```
CREATE TABLE `maidsintrusion` (
# id of record
 `id` int() NOT NULL AUTO_INCREMENT,
# intrusion address
 `address` varchar(25) NOT NULL,
# intrusion location
 `location` varchar(25) NOT NULL,
# intrusion date
 `intrusiondate` varchar(30) NOT NULL,
# person’s name reporting intrusion to authorities
 `reportingperson` varchar(25) NOT NULL,
 # reporting person’s contact phone
 `contactphone` varchar(25) NOT NULL,
 # reporting person’s contact email
 `contactemail` varchar(25) NOT NULL,
   PRIMARY KEY (id)      # Make the id the primary key
);
```

10.5 Establishing Database Connectivity

Accessibility to the MAIDS database was accomplished as follows:

1. The Python3 dbinsert() function imports the mysql.connector and Error modules to handle the connection to the database and any errors that might result during the connection try. (i.e. import mysql.connector and from mysql.connector import Error)
2. Then, connection parameters are defined as follows:
    a. Host Ip Address (host=192.168.xxx.xxx)

    b. Database Name (database=maids1)

    c. Username (dbusername=clauxxxxxxxx)

    d. Database Password (dbpassword=xxxxxxxxxx)
3. Finally, the function configures a connection to the database server on MAIDS by using the following command:

    a. connection = mysql.connector.connect(host,database,user,password)
4. A message is displayed on the Android application if the connection is successful or not.

10.6 Inserting Data into Database

Once the connection to the database is made, the data from the intrusion is gathered with the following statement into the db_data variable:

a. db_data = (address, location, intrusiondate, reportingperson, contactphone, contactemail)

The data form the intrusion is then formatted for insertion into the database using a parametrized insert statement:

a. mySql_insert_query = """INSERT INTO maidsintrusion (address, location, intrusiondate, reportingperson, contactphone, contactemail)VALUES (%s, %s, %s, %s, %s, %s) """

Once the parametrized statement is constructed, the data is inserted into the database with the following command:

a. cursor.execute(mySql_insert_query, db_data)

Finally, the information is committed to the database and the connection is closed with the following commands:

a. connection.commit()
b. print("Record inserted successfully into MAIDS-DB")
c. cursor.close()
d. connection.closed()

10.7 Testing Database Connectivity

Testing connectivity to the MAIDS' MySQL database employed a simple TCP connection (Telnet) using putty software. Putty is a free and open-source terminal emulator application that supports several network protocols (i.e. SCP, SSH, Telnet, rlogin, and raw socket connection). The advantage of using telnet is that is extremely simple. There are no configuration files to modify and no authentication. It either makes the connection or does not. The database was accessible. Therefore, further successful testing was done with the MAIDS' Android application.

10.8 Security Considerations

Risks are inherent in any application requiring users to supply input (i.e. usernames, passwords, personal information, etc.) Database attacks resulting in data breaches may come from SQL virus attacks or from employee misuse. Successful attacks can lead to the theft of thousands (sometimes millions) of records containing valuable data (i.e. personal information, credit card, financial, healthcare, etc.) Therefore, the main purpose behind any cyber-attack is to get access to a database located on a server and it is therefore imperative that the security of the database server is strengthened. The strengthening of the database depends on database and network security, operating system hardening and physical security. The MAIDS server database was secured in the following ways:

1. Deploying a strong password policy execution – A strong password policy is the front line of defense to confidential user information. A password policy is a set of rules which are created to improve computer security by motivating users to create dependable, secure passwords and then store and utilize them

properly. MAIDS implements a strong password policy based on the following criteria:

a. At least 8 characters long
b. Should not contain personal information (i.e. username, person name, etc.)
c. Should not contain dictionary words.
d. Should contain uppercase letters, lowercase letters, numbers, and characters

2. Discard Default Users and Demo-test Databases – Default users and demo databases are public record. Therefore, MAIDS removed default users and demo databases so that an attacker is prevented from collecting details on the database and user information.
3. Implement MySQL Enterprise Firewall – MySQL Enterprise Firewall guards against cyber security threats by providing real-time protection against database specific attacks. MAIDS implements the MySQL Enterprise Firewall to protect data by monitoring, alerting, and blocking unauthorized database activity without any changes to the application. The system variable mysql_firewall_mode was set to ON in order to implement the MySQL Enterprise Firewall (i.e. mysql_firewall_mode=ON)
4. Changed the Admin User Name – For example, the name "admin" is not very secure, because it's easier to hack via brute force. Therefore, for additional security, MAIDS changed the admin username.

5. Restricted User Privileges – Access and privileges were limited to the database user’s needs. This will help in preventing data loss even after an exploit attempt.
6. Disabled Public Network Access to the Database Server – All public network access to the database server was blocked.
7. Implemented iptables and Malware Scanners – iptables (Linux-based firewall) was setup on the operating system and malware scanners were frequently ran on the database server.
8. Changed the port for the outside – Changing the database default port (3306) will stop bots that target said port from directly attacking that specific port.
9. Set MySQL SSL – SSL stops transmitting data in clear between the client and the server.
10. Only allowed access from certain IPs -- This ensures that only certain IP’s will be able to connect to the database server (using iptables)

10.9 Unit Testing

Unit testing is a software testing method by which individual units of code are tested in isolation. The purpose of unit testing is to isolate the smallest testable parts of an API and verify that they function properly in isolation. A unit test can verify different behavioral aspects of the system under test (SUT), but mainly it verifies that the SUT produces the correct results. Unit testing ensures the whole application or system works as expected by ensuring the critical application and device capabilities, such as user logins, network connections, database reachability, Internet services connections, and all the critical hardware behaves as expected.

The benefits of writing and performing unit-tests are as follows:

a. Makes the coding process more Agile.
b. Issues can be found very early and can be resolved.
c. Refactor code or upgrade system libraries and make sure modules work correctly.
d. Provides documentation of the system.
e. Simplifies the debugging process.
f. Helps reduce the cost of bug fixes.
g. Increases confidence in changing/ maintaining code.
h. Codes are more reusable.

The table below lists the unit testing results for the MAIDS device.

TABLE 8 UNIT TESTING RESULTS FOR MAIDS PROJECT.

Component Tested (SUT)	Expected Behavior	Actual Test	Expected Result
Motion Module	Motion is sensed when object moves in front of the motion module	a. Person moves in front of MAIDS motion sensor module b. Motion test Python3 code is ran	Motion is sensed and appropriate response is triggered.
Sound Module	Sound is sensed when object moves in front of the sound module	a. Person recreates a loud sound in front of MAIDS sound sensor module b. Sound test Python3 code is ran	Sound is sensed and appropriate response is triggered.
LED Module	LED module lights up green or red depending on sensing situation	a. Motion/sound sensors are not triggered. b. Motion/Sound sensors are triggered	d. Motion/sound sensors triggered LED module lights up green.

		c. Run LED module Python3 code.	e. Motion/sound sensors triggered LED module lights up red. f. Module runs and green/red LED’s light up.
Camera Module (No python3 code was ran)	Camera module triggered by room intrusion	Trigger sensors (motion/sound) one-at-a-time to trigger the camera module	Motion/sound sensors triggered and photograph is produced.
Switch Module (No Python3 code was ran)	Switch module powers MAIDS device when switched ON and powers MAIDS device down when switched OFF	a. Press switch to ON position b. Press switch to OFF position	c. MAIDS’ device powers up. d. MAIDS’ device powers down.
Database Module	Database reached via Internet returns	“Retrieve database data” and “Retrieve intrusion photo”	a. Database data is retrieved and displayed on

	recorded data and intrusion picture	buttons clicked in Android software activity (Python3 code ran when button clicked)	Android application screen. b. Intrusion photo is retrieved and displayed on Android application screen.
Enterprise Network Connection	Connect to any enterprise network defined in wpa-supplicant file	a. Connect to home network b. Connect to Humber College Network c. Connect to Tim Hortons' network	Connection to all networks defined in wpa-supplicant file achieved.
Internet Services Connection (Twilio, PushNote, eMail, phone call)	a. Receive Twilio SMS message when sensors triggered. b. Receive PushNote	a. Trigger sensors to receive SMS message. b. Trigger sensors to receive	When sensors were triggered the following occurred: a. SMS message received.

	message when sensors triggered. c. Receive email message when sensors triggered. d. Receive phone call when sensors triggered.	PushNote message. c. Trigger sensors to receive email message. d. Trigger sensors to receive phone call.	b. PushNote message received. c. Email message received. d. Phone call received.
Android Application Login	User inputs username and password and log into the Android application	Input the username and password into Android application Login Activity	Entered username and password allows login into Android application
Control activity – Activate MAIDS	Remotely activate MAIDS device	Click on “Activate MAIDS” button in Android application	MAIDS device is remotely activated
Control Activity – Deactivate MAIDS	Remotely deactivate MAIDS device	Click on “Deactivate MAIDS” button in Android application	MAIDS device is remotely deactivated

10.10 Production testing

In production testing, the tester is not concerned with the executable code. Instead, the tester is concerned with the need to verify the output of the MAIDS device based on given user requirements against the expected output. The prime objective of MAIDS production testing is to check the functionalities of the system. These tests, check the Android application, device hardware (Raspberry Pi 4 and custom-made PCB board), and networking infrastructure, from the front end UI to the back-end database systems. In that sense, the MAIDS production tests are also a form of integration testing, ensuring that different components are working together as expected.

The MAIDS' production testing environment was designed to measure and monitor the following:

1. Measure the application's performance in real-time.
2. Monitor the application in real-time to detect network failure and weak connections.
3. Monitor the API responses at peak traffic.
4. Detect bugs which typically go unnoticed.
5. Detect possible point(s) of failure.
6. Help maintain the high quality of the device and application.
7. Produce a reliability statistic.

Consequently, individual, as well as an integrative, production tests were performed and the results tabulated below.

TABLE 9 PRODUCTION TESTING RESULTS FOR MAIDS PROJECT.

Test No.	Components tested	Expected Results	No. of Trials	Success Rate (%)	Failure Rate (%)	Actual Results
1	PCB board, sound sensor, LED module, audible messages, network connection, Android application	Android application remotely triggers PCB board and sensor module exchange intrusion information that triggers the LED module and audible messages.	100	98	2	PCB board, sound sensor, LED module and audible, network connection and Android application performed as expected.
2	PCB board, motion sensor, LED module, audible messages,	Android application remotely triggers PCB board and sensor	100	99	1	PCB board, motion sensor, LED module, audible, network connection and

	network connection, Android application	module exchange intrusion information that triggers the LED module and audible messages.				Android application performed as expected.
3	PCB board, LED module, network connection, Android application	Android application remotely triggers PCB board and LED module information exchange that triggers the LED module (green or red)	100	100	0	PCB board, LED module, network connection and Android application performed as expected.
4	PCB board, network connection,	Android application triggers PCB	100	100	0	PCB board, network connection and

	Android application (MAIDS remote activation)	board to activate MAIDS device.				Android application performed as expected.
5	PCB board, network connection, Android application (MAIDS remote deactivation)	Android application triggers PCB board to deactivate MAIDS device.	100	100	0	PCB board, network connection and Android application performed as expected.
6	PCB board, motion sensor, network connection, Twilio services (SMS message and phone call).	PCB board and motion sensor module exchange intrusion information that triggers the LED module,	100	100	0	PCB board, motion sensor, network connection and Twilio service (SMS and Phone call) performed as expected.

		audible message, and Twilio service.				
7	PCB board, motion sensor, network connection, PushNote service (mobile pushnote).	PCB board and motion sensor module exchange intrusion information that triggers the LED module, audible message, and PushNote service.	100	100	0	PCB board, motion sensor, network connection and PushNote service (mobile pushnote) performed as expected.
8	PCB board, motion sensor, network connection,	PCB board and motion sensor module exchange intrusion	100	100	0	PCB board, motion sensor, network connection, email service and camera

	email service and camera.	information that triggers the LED module, audible message, and emails intrusion photo and data to recipient.		performed as expected.

11.0 Results and Discussions

11.1 General Project Outcomes

By no means is the MAIDS system perfect and further improvements are required. However, the project managed to accomplish the outcomes described below.

11.1.1 Surveillance Capabilities

Motion and sound sensor devices within MAIDS cause the initiation of a local audible/visible alarms and notification events.

11.1.2 Multi-channel Alerts

Transmission of alarm alerts to home/business owner, police department or central monitoring station via email (with picture), Android Push Notifications, SMS Messaging and phone calls.

11.1.3 Sound Sensing Unit

A sound sensing device that has the ability to detect continuous attack noises in the audio frequency range up to 10 kHz.

11.1.4 Motion Sensing Unit

A motion sensing device that has the ability to detect distinguish between object movement and human movement, cover a motion cone of 110°, distance of up to 7 meters within operating temperature from -20° to +80° Celsius and low power consumption of 65 mA.

11.1.5 Signal Processing Circuitry

Provides separate signal processing circuitry for independent sensing microphone and PIR motion sensor.

11.1.6 Intrusion Alarm System

Intended for use in intrusion alarm systems to provide premise-protection (home/business) of spaces and other secure areas

11.1.7 Visual Capabilities

The camera takes a photographic record of the event and intruder when the alarm is activated providing a caption containing time of entry, place of entry and address of home/business. Photographic/video record is included in email message to the owner and can help law enforcement track down potential criminals or trespassers.

11.1.8 EMI Resistance

A sound sensing device that will not enter the alarm state when subjected to moderate levels of radiated electromagnetic fields and conducted interference.

11.1.9 Reliability of the device

MAIDS performed reliably in 50 out of 54 trial runs.

11.1.10 Control Unit

Raspberry Pi 4, Python Code and Sound/Motion sensing devices that provides the electronic circuitry to process the signal from the sensor and initiate an alarm signal when attack noises are detected.

11.1.11 Secure Mode (All-Safe Mode)

Sound/Motion sensing system where all sensors and control unit are active and ready to respond to attack sounds and motions.

11.1.12 Strategic Placement of Components

Electronic alarm sensors strategically placed so that they can monitor conditions that require security alerts.

11.1.13 Enclosure Protection

Designed MAIDS case to protect equipment from damage.

11.1.14 Web-based Intrusion Database Log

Designed and implemented a web-based intrusion database log developed with PHP and MySQL.

11.2 Project Issues/Challenges

A few issues and challenges were encountered during the design and implementation of the MAIDS project which gave way to possible future improvements.

11.2.1 Project Issues

1. Sensitivity Adjustments: Adjusting the sensitivity of the motion sensor can be tricky with the sensor's potentiometer.
2. False Alarms: Improper installation of the device (common traffic area/improper heights) can lead to a false detection caused by the movement of objects such as pets, blinds, and curtains within the range of a motion detector. The implementation of AI in MAIDS, as discussed below, can reduce false alarms considerably.
3. Overlapping Traces: PCB board design has to take into consideration short-circuits created between overlapping traces. Therefore, overlapping traces should be placed in different layers.
4. Sensor Threshold: Adjusting the sensor's threshold of allowed movement so that small movements in the room from events such as blind movement do not constantly set the alarm system off.

11.2.2 Project Challenges

1. Environmental Impact: New standards and regulations require electronics designers and manufacturers to consider the environmental impact of a product's entire life cycle. MAIDS tried to consider every aspect, from the manufacturing process, chemicals and tools used, to consumer energy use and disposal of the product.
2. Stringent Quality Control Methods: For MAIDS, it is important to produce good quality products. Consumers want electronic products that operate the way they should. Therefore, MAIDS implemented strict quality control measures to ensure the consistent quality of all the products produced.
3. Lack of skills: A lack of appropriate skills and an understanding of embedded technology can become an important issue in the design process. MAIDS strived to understand the embedded technology used (Raspberry Pi 4) and the associated GPIO pins, as well as, learn and apply the skills (i.e. CAD design, 3-D printing, soldering, etc.) required by the project.
4. On-time Delivery of Project: There is an inherent pressure to deliver projects quickly. Therefore, lead times become a primordial concern.

11.3 Project Lessons

11.3.1 Design Lessons

In order to minimize the design risk, MAIDS minimized the design complexity. Using electronic modules, as MAIDS has, for instance, was the common way to reduce the design complexity and risk.

11.3.2 Soldering Lessons

The following are lessons learnt during the soldering process that are important to follow.

1. Set the tip temperature to the temperature appropriate to the solder alloy being used.
2. Use lead-free solder; healthier for the people working on the project and for the environment.
3. Place component and fix two opposing corner pins.
4. Clean the solder-well tip on a sponge.
5. Do not over tin the tip with solder
6. Remove flux residue if necessary.

11.3.3 Breadboard Building Process Lessons

The following are lessons learnt during the breadboard building process that are important to follow.

1. Breadboard circuits should only be used for designing and testing circuits outside of cases and housings, before you move on to a soldered version.
2. Tools are not needed most of the time, however, it is helpful to have a pair off tweezers or needle-nose pliers to handle some smaller components.
3. Component insertion into the breadboard must be done by pushing the component leads into the breadboard holes straight down and trimming them if they are not the right length.
4. Always pay attention to component and cable management in general, especially when it comes to arranging jumper wires. Otherwise, one will end up with a tangled, disorganized board.
5. Jump wire kits can provide the various lengths and color-coding options that will help organize a project as the project gets more intricate.
6. Do not wire individual components directly to power source. Instead, use the power rails.
7. A digital multimeter should be used often to check connections between holes and rails.

11.3.4 PCB Board Lessons

The following are lessons learnt during the PCB board design process that are important to follow.

1. Draw and overview plan of where the different circuit components will be located.
2. Allow adequate board area for the circuit.
3. Do not place traces at right angle.
4. Ensure same orientation while placing components.
5. Keep power and control ground separate from each other.
6. Allow sufficient space for cooling around hot components.
7. Consider track size for lines carrying current.

11.3.5 Project Documentation Storage Lessons

Perhaps the most important lesson of all regards the project's documentation storage. Storage and backup of the project's documentation should be paramount at any stage. These documents must be kept in a secured storage area and include availability redundancy.

While document storage might seem somewhat inconvenient, costly, and rather time-consuming, the loss of a project's documentation will cost a great deal more, not only to the project designers, but for the partners, clients and staff as well. Using a secure document storage facility is the safest, easiest and most cost effective way to ensure proper storage of documents. Therefore, MAIDS made use of OneDrive by Microsoft and GitHub to secure and store all documentation pertinent to the project. There is nothing more frustrating and catastrophic than losing all the documentation about a project due to human error, data loss or corruption, theft, sabotage or malware attack.

11.3.6 Technical Lessons

During a project’s life cycle, it is important to ensure consistent and timely progress reporting. Furthermore, the accuracy of the information and report data must be guaranteed. Finally, one needs to anticipate and exploit evolving technology.

11.3.7 Organizational Lessons

Organizational lessons include the need to clarify project and functional roles and responsibilities and understand the skills required. Also, we need to heed and measure the capacity to develop the project successfully. Finally, it is important to establish a consistent reporting process that can be used to improve decision making.

11.3.8 Marketing and Sales Lesson

Creating something great is not the path to success. Creating something great and being able to market and sell it is the key to success.

11.3.9 Final Lesson

The key to business success is always to build, test, learn, and repeat.

11.4 MAIDS Project Proposed Improvements

In order to further improve the MAIDS system, the following subsequent improvements should be undertaken:

1. Tamper Proof: Provide some type of alert if the alarm device has been tampered with or opened.
2. Protective Covering: Provide protective covering for surface-mounted contact switches, wire connections, and wire distributions. These protective coverings must be strong enough to withstand damage due to collisions and bumps.
3. Power Supply/Batteries: Alarm sensors need a power supply that cannot be interrupted. Backups and/or batteries will be required.
4. 2-way calling: Configured to allow two-way calling with your alarm company. This will allow you to speak to your security system provider without picking up the phone.
5. Away Mode: Provide and automatic Away Mode - the system assumes that you are out of the house, and will therefore enable all sensors between certain daily hours.
6. Central Monitoring: Provide a central monitoring station connected to the home/business security system for action in times of emergency.
7. Keypad Authentication: Addition of a keypad for authentication, arming/disarming of MAIDS.
8. Add-ons and Integrations: Allow users to create custom zones using a combination of sensors and cameras.

9. Future Technology Implementation: Concerns about future technology developments, including component obsolescence, can render an entire product difficult to sell. The MAIDS project tried to mitigate component obsolescence by incorporating the latest available modules into its construction.
10. Future Programming Implementation: In order to thrive in there is an inherent need to keep on top of programming developments. MAIDS should be updated to use the Paramiko library which implements the SSH2 protocol as an alternative to SSL for making secure connections between python scripts. All major ciphers and hash methods are supported. SFTP is also supported by Paramiko.
11. Emerging Artificial Intelligence (AI): Emerging technologies like AI can disrupt industries as well as providing new opportunities. For example, MAIDS video camera should be updated with AI with facial recognition functionality to identify whether the moving object is an intruder or a home member. This will provide a new mechanisms that improves accuracy in detecting intrusion into the home while reducing the chance of false alarms.

11.5 Project Best Practices

In order to have MAIDS function properly, the following best practices should be implemented upon deployment of the system.

1. Optimal Placement: The optimal place to install MAIDS is in a corner, so the 90 degrees of coverage run along each wall, effectively covering the maximum amount of space.
2. Optimal Angle: Motion detectors take a longer time to react to someone walking in a straight line directly towards the motion detector's lens, therefore, motion detectors are best suited to detecting movement made across the room, parallel to the lens.
3. Minimum Height: Install motion sensors at a height between 7 and 8 feet above the ground pointing downwards at an angle to cover the room.
4. Pet Proofing: Pet-proof motion detectors require a minimum of six feet between the motion detector and the animal to be effective so base your height placement on the height of your cat/dog at his tallest point when standing or jumping, depending on temperament.
5. Designated Surveillance Areas: Confine pets to areas that are not covered by your motion sensors while you are away.

12.0 Conclusions

Mass production (also referred to as flow production, repetitive flow production, series production or serial production) is the process of manufacturing large quantities of a product employing computerization technology. In mass production, automation is used to achieve high volumes, organize material flow, and control quality standards. Mass production "…provides a rigorous way to monitor production resulting in lower costs, use of fewer resources, high levels of efficiency, quick assembly, prompt distribution and marketing creating a competitive advantage and higher profits" (Banton, 2020). Unfortunately, there are also disadvantages such as a significant upfront investment of time, money and resources. Therefore, it is imperative that a balance be obtained.

12.1 Before Mass Production Starts

It is recommended, before the MAIDS device is mass produced, that the following initial steps be taken into consideration and implemented:

1. Market Research: Attending industry trade shows will provide an idea as to the viability for product sales. It will provide an idea on which side of the market spectrum the MAIDS device will be placed successfully.
2. Securing Some Early Funding: It is imperative that production costs be considered. Providing the necessary capital for the first production run may include one or a combination of the following: personal wealth, find and investor or take out a bank loan.
3. Implement a Non-Disclosure Agreement (NDA): Implement an NDA (non-disclosure agreement) in place: Have people sign it to prevent people that know details of the MAIDS device from stealing the idea or telling others about it.
4. Learn About Certifications Required: In Canada, all products are regulated by some sort of federal, state and/or local agency that prevent the manufacturer from breaking any rules. It is imperative to learn about the certifications required for the particular product before starting the project. Implementing national standards assure consumers that the MAIDS device meets consistent and uniform rules but might prove too costly or unattainable.
5. Licensing: One major decision to make is whether to produce and sell the product yourself or license the idea to a company with the means and experience to handle it. The company handles everything – the manufacturing, marketing, distribution – and then pays you royalties based on sales. No upfront investment is required.

6. Setup a Formal Business Structure: A sole proprietorship is a business that is owned by one person and it is the easiest, least expensive type of business to start. It is advisable that MAIDS start with a sole proprietorship, but switch to an LLC before beginning to sell it.

12.2 Mass Production Cost Considerations

Most entrepreneurs drastically underestimate all of the costs required to develop, scale and manufacture a new electronic hardware product. This is one of the main reasons so many businesses ultimately fail. Therefore, one must give careful consideration to the costs outlined below.

12.2.1 Development Costs

Development costs for most hardware products are broken down into three categories: the electronics, the plastic and other mechanical parts, and the retail package. The electronics does all of the magic, the plastic and mechanical parts hold the product together, and the retail package protects and sells the product.

12.2.2 Electronics Cost

The electronics will usually be the most complex and expensive part of your product to develop, unless, one does their own product design. Prototyping the electronics is divided into two steps: production of the blank Printed Circuit Board (PCB) and soldering of all the electronic components onto the PCB. The PCB is what holds and connects all of the individual electronic components. So in most cases, it is bets to use standard through-hole vias. In the MAIDS project, the ratio of electronic prototyping costs to board assembly costs were 1:2. So, it is recommended that small prototyping quantities are produced initially and then potentially increase the quantity through each iteration. Once functionality has been confirmed and bugs have been resolved it is recommended that prototype quantities be increased and samples shared with investors and potential customers.

12.2.3 Enclosure/Mechanical Development Cost

MAIDS will require an enclosure which is made of plastic. The appearance and ergonomics of the MAIDS device are critical for the product, and in turn, will increase design costs. It is recommended that 3D printing technology be used to bring down the cost of creating plastic prototypes, it provides fast turn-around time (less than 24 hours) and lower costs when small volumes are required, as is the case during the design phase of the project. Therefore, it is recommended that a 3D printer be purchased as the most cost effective strategy.

In regards to mass production, it is recommended that injection molding be used because it is the most economic option, due to mechanisms of economies of scale. Injection molding refers to the "…process of creating a components by injecting under pressure melted material into a die. The material fills the hollow cavities of the mold and when it cools it solidifies, taking the form of the die." (Varotsis, 2020) Injection molding can yield very high production rates.

12.2.4 Scaling Costs

From prototypes to large volume production there is a big difference. Mass production must take into account scaling costs but oftentimes it is one of the most underestimated steps in launching a new product. Scaling costs include: certification costs,

12.2.4.1 Manufacturing Setup Costs

For MAIDS, it is recommend to start the manufacturing process with a local manufacturer within Canada. When manufacturing volumes approaches more than ten thousand pieces, then migrate to an Asian manufacturer. It is also recommended that help from experts in Asian manufacturing is sought when the time comes to shift production to an Asian country. Consequently, it is recommended that MAIDS use multicultural marketing resources, such as Expert's Directory (a resource that features a range of ad agencies, marketing, research, communications and PR firms, media companies, consultants and others), who are experts in outreach to all Asian segments. See appendix X for a partial list of Asian Market experts.

12.2.4.2 Certification Costs

Products require multiple certifications before they are release into the Canadian and American markets. Certification costs may be a few thousand dollars or as high as a few tens of thousands of dollars. It depends largely on the product and to a large extent how any wireless features are implemented. The main certifications required by the MAIDS system include:

1. FCC (Federal Communications Commission) Certification: It is required of all electrical products sold in the U.S.A. All electrical products radiate electromagnetic energy so governments want to ensure they do not interfere with RF communication. MAIDS is classified as an intentional radiator product (they intentionally radiate radio waves) and therefore will cost about ten times more than non-intentional radiator products. In order to reduce costs pre-certified modules (an electronic circuits developed to perform a single function and to be incorporated into other designs) will be used. At higher production volumes wireless module should be transitioned to a custom wireless design to increase profit margins.

2. Underwriters Laboratories (UL) or Canadian Standards Association (CSA) certification is required for any product that will be sold in the USA or Canada that plugs directly into an AC electrical outlet. This cost can be removed by selling the product online. However, UL and CSA certifications will be required when sold in large retail chains. The recommendation for MAIDS is to start with online sales, wait to see if the product sells well, and if it does, move to retail stores (a step which will then require certification costs, not before).
3. CE certification is necessary for electrical products that will be sold in the EU (European Union). It is similar to a combination of FCC and UL certifications. Since California is a huge market, MAIDS should be CE certified.
4. RoHS certification guarantees that the product is free of lead and is necessary for any electrical product that will be sold in the European Union (EU) and California. MAIDS should be RoHS certified to instill on the customer the company’s regard for the environment.

12.2.4.3 Landing Costs

No doubt about it, the landed production cost is the most important cost for the MAIDS project. The landed production cost is the total cost to produce and transport a single unit to the warehouse. MAIDS will always be striving to reduce this cost so the company can ultimately achieve greater profits. For most products, one can estimate the suggested sales price to be 3-5 times the landed production cost. The landed production cost will definitely be the most important cost since it determines the profit, sales price and inventory cost of the MAIDS system. Some of the many costs that make up the landed production cost include:

1. Electronic Costs
2. PCB production and assembly
3. Injection molded plastic parts
4. Final Product Assembly
5. Testing
6. Packaging
7. Returns
8. Freight
1. Duties

12.3 Retail Package Development

The retail package is just as important as the product itself. Sometimes it is even more important. One can have the greatest product in the world but if the retail package does not convey this point to the customer, the product will not sell. Since the MAIDS device is small it is recommended that clamshells be used. Clamshells consists of two parts: a custom shaped plastic piece to hold and protect the product, and a cardboard artwork piece to convey the sales message. A reduction in cost can be achieved by using a custom molded blister (the part of the clamshell that custom fits over your product). Blister packaging is an inexpensive option for creating packages that are durable, transparent, and tamper proof. In addition, the clamshell insert card can also be printed on regular paper prototypes to reduce costs further.

12.4 MAIDS Retail Price Determination

Choosing MAIDS retail price is a very important consideration that should be addressed promptly. If the price is too low, the MAIDS project would not make sufficient profit. It the price is too high, the MAIDS device will not sell. The strategy for the MAIDS device is to set the price high so that if need be the price could be lowered later on. This is a tricky situation because if the price is too high sales might not recover even after lowering the price. In order to estimate the optimal price for the MAIDS device it is imperative to know how much the device costs to make. Calculating the cost of the device is called Cost-of-Goods-Sold (COGS). In order to calculate the COGS all the costs to produce the MAIDS device are added up. These costs include the following:

1. Electronic components (sensors, connecting wires and pins)
2. Production of your Printed Circuit Board (PCB)
3. PCB Assembly (soldering of components onto the PCB)
4. Enclosure Plastic Parts (injection molded plastic)
5. Product assembly
6. Product testing/Quality control
7. Import and/or export duties and taxes
8. Warehousing and logistics

Estimation of the abovementioned costs allows one to decide if the MAIDS device will be profitable before spending money in development.

12.5 Product Positioning

The MAIDS device will target the elderly market. This demographic market includes elderly families that want a trained professional installing their alarm system in a large home with a substantial number of doors and windows and a secluded garage door. At the same time, one must not alienate the millennial and younger generation since they are the homeowners of the future.

12.6 Distribution strategy

Initially, the best distribution strategy for the MAIDS device is web-based. In other words, sell the MAIDS device via a website. Selling the product in this manner will increase profits more than selling it through a retail store because one can charge a lower price; the profit margin is greater without a retailer taking a cut. Eventually, MAIDS is likely to move up to selling in retail outlets and through multiple distribution channels.

A Family Security Blog (FSB) can greatly assist in building brand awareness and product audience. Within the blog, one can provide valuable free content, such as, MAIDS product information, discussions regarding the alarm industry and areas of expertise. The point of the blog is to present the MAIDS project as an expert in the field, collect email addresses for advertising and selling purposes, product validation and obtain feedback on the product.

12.7 Social Cost of Mass Production

There are political, economic and social costs that need to be taken into account as a responsible MAIDS device manufacturer. These costs include:

1. Misuse of natural resources
2. Pollution generated by factories and transport of goods
3. Pollution generated by plastic
4. Electronic Waste Created
5. Greenhouse gases generated
6. Water Pollution

These factors must be given careful consideration because of the political, social and economic problems that might arise.

Therefore, taking a responsible approach to MAIDS manufacturing the MAIDS project assumes an extended producer responsibility (EPR). ERP is a "…practice and a policy approach in which producers take responsibility for management of the disposal of products they produce once those products are designated as no longer useful by consumers. Responsibility for disposal may be fiscal, physical, or a combination of the two." (Surak, 2020)

12.8 Minimizing Risk

The MAIDS device is successful if the costs and time put into it are minimized from the very beginning. The MAIDS project minimizes risk by employing the following criteria:

1. Minimize complexity by using modules (i.e. sensors, LED module, etc.)
2. Use standardized communication protocols such as Wi-Fi or Bluetooth that require standard electronics
3. Use pre-certified and tested modules
4. Minimize Certification Costs
5. Review the MAIDS device with an independent engineer to mitigate issues.
6. Review it to make sure that there aren’t any issues.
7. Simplify the Enclosure
8. Focus on minimizing the total cost you have to spend upfront

13.0 References

Banton, Caroline. (February, 2020). Investopedia: Mass Production. Retrieved from https://www.investopedia.com/terms/m/mass-production.asp on February 12, 2020.

Bittele, Electronics. (2020). PCB Trace Width Calculator. Retrieved from https://www.7pcb.com/trace-width-calculator.php on February 12, 2020.

Brinks. (2019). Retrieved from https://help.brinkshome.com/hc/en-us/articles/360006895212-Faster-response-with-ASAPer?kbid=117104, September 2019.

Canada, S. (2002). Breaking and Entering in Canada - 2002. Retrieved from Statistics Canada: http://www.publications.gc.ca/site/archivee-archived.html?url=http://www.publicatio on September 2019.

Canada, S. (2018). Incident-based crime statistics, by detailed violations, Canada, provinces, territories and Census Metropolitan Areas. Retrieved from Statistics Canada Table 35-10-0177-01 : https://www150.statcan.gc.ca/t1/tbl1/en/tv.action?pid=3510017701&pickMemb on September 2019.

Canada, S. (2020, September). Preventing break-ins. Retrieved from SGI CANADA. 2020: https://www.sgicanada.ca/news?title=preventing-break-ins

Christenson, P. (2017, December 16). RDBMS Definition. Retrieved from https://techterms.com on February 12, 2020.

Electronics, L. L. (2008). ProtoLaser S: Operation manual 2.0, English. Bohn: LPKF Laser & Electronics AG. Retrieved from LPKF Laser & Electronics AG. (2008). ProtoLaser S: Operation manual 2.0, English. Page35.

Electronics, L. L. (2020). Manual Version 0.9, English. Germany: LPKF Laser & Electronics AG.

Elektrotanya. (2020, January 12). Model WES51 Electronic Soldering Station. Retrieved from Model WES51 Electronic Soldering Station: https://elektrotanya.com/weller_model_wes51_electronic_soldering_station.pdf/download.html on January 15, 2020.

Geometries, O. (2010). Objet30 3-D Printer System: User Guide, English. Page 6. Germany: Objet Geometries Ltd.

Kester, Walt. (2016). Analog Dialogue: Breadboarding and Prototyping Circuits, Vol. 50. Retrieved from https://www.analog.com/en/analog-dialogue/studentzone/studentzone-november-2016.html# on February 10, 2020.

King, R. (2019). The secret to stopping break-ins. Toronto: MoneySence Magazine. Retrieved September 22, 2019, from https://www.moneysense.ca/spend/real-estate/the-secret-to-stopping-break-ins/.

Malinen, J. (2020, February 8). wpa_supplicant(8) - Linux man page. Retrieved from die.net: https://linux.die.net/man/8/wpa_supplicant on February 8, 2020.

Medri, K. (2020). Course Modules: Welcome. Toronto: Humber College. Retrieved January 8, 2020, from https://learn.humber.ca/webapps/blackboard/content/listContent.jsp?course_id=_148549_1&content_id=_7882456_1

OAECTT. (2017). Technology Report Guidelines Revised March 2017. Toronto: Ontario Association of Certified Engineering Technicians And Technologists (OACETT).

Rembert, Ludovic. (Feb. 22, 2020). {rivacy Canada: Best Home Security Systems (2020). Retrieved from https://privacycanada.net/best-home-security-system/- on February 12, 2020

Radian Thermal Products, Inc. (2020). Aluminum Heatsinks. Retrieved form https://www.radianheatsinks.com/aluminum-heatsink/ on February 12, 2020.

Rajaniemi, J. (2012). Measuring and Defining Lead Time in a Telecommunication Production. Retrieved from leadtimes.org: http://www.leadtimes.org/

Safety.com. (2020, January 22). The Best Home Security Systems of 2020. Retrieved from Safety.com: https://www.safety.com/best-home-security-systems/

Seymour, R. (2015, November 1). Organization & Cleaning - Home security: 10 ways to protect your home from intruders. Retrieved from Canadian Living: https://www.canadianliving.com/home-and-garden/organization-and-cleaning/article/home-security-10-ways-to-protect-your-home-from-intruders?kbid=117104

SparkFun. (2020). PCB Basics. Retrieved form https://learn.sparkfun.com/tutorials/pcb-basics/all on February 10, 2020.

Surak, Sarah M. (December 6, 2018). Extended Producer Responsibility: Environmental Practice And Policy. Retrieved on March 10, 2020 from Https://Www.Britannica.Com/Topic/Extended-Producer-Responsibility.

Trotec. (2020, January 19). Operation Manual Trotec Job Control. Basic, Advanced, Expert. Retrieved from Trotec: Operation Manual Trotec Job Control. Basic, Advanced, Expert: https://www.troteclaser.com/fileadmin/content/images/Contact_Support/Manuals/JobControl-Manual-EN.pdf.

Varotsis, Alkaios. (2020). 3D Hubs: 3D Printing vs. CNC Machining. Retrieved on February 23, 2020 from https://www.3dhubs.com/knowledge-base/3d-printing-vs-cnc-machining/#intro.

Woodall, M. (2019, September 14). Canadian Crime Rates Burglary & Home Invasion: A Real Threat. Retrieved Septrember 14, 2019, from SecureHouse.ca: http://www.securehouse.ca/canadian-crime-rates-burglary-home-invasion-toronto.html

Woodall, M. (2019, December 18). Top 5 Best Home Security Systems in Canada 2020. Retrieved from Reviews.org: https://www.reviews.org/home-security/best-security-systems-canada/

Glossary of Terms

AC – Alternating Current.

API - Application Program Interface.

Bluetooth - standardized protocol for sending and receiving data via a 2.4GHz wireless link.

Breadboard –a construction base for prototyping of electronics.

BOM – Bill of Materials.

DC – Direct Current.

CLI – Command Line Interface.

CPU – Central Processing Unit.

CSA – Canadian Standards Association.

CSS – Cascade Styling Sheet.

Decibel - relative unit of measurement corresponding to one tenth of bel.

DRC - Design Rule Check.

Ethernet - family of computer networking technologies commonly used in local area networks, metropolitan area networks and wide area networks.

Embedded System - a combination of computer hardware and software, either fixed in capability or programmable, designed for a specific function or functions within a larger system.

Firmware - permanent software programmed into a read-only memory.

FR-4 - NEMA grade designation for glass-reinforced epoxy laminate material.

Gerber File - open 2D binary vector image file format.

Gigabit - one billion bits, or 1,000,000,000 (that is, 10^9) bits. It's commonly used for measuring the amount of data that is transferred in a second between two telecommunication points.

GHz - Short for gigahertz, is a unit of measurement for AC (alternating current) or EM (electromagnetic) wave frequencies equal to 1,000,000,000 (one billion) Hz (hertz).

GPIO - general-purpose input/output pins on a Raspberry Pi.

GPU - graphics processing unit.

HDMI - High-Definition Multimedia Interface is a proprietary audio/video interface for transmitting uncompressed video data.

HEVC - high-efficiency video coding.

HTML – Hypertext Markup Language

IDE – Integrated Development Environment.

IEEE 802.11 - family of specifications developed by the IEEE for wireless LAN (WLAN) technology. 802.11 specifies an over-the-air interface between a wireless client and a base station or between two wireless clients.

Internet - a global computer network providing a variety of information and communication facilities, consisting of interconnected networks using standardized communication protocols.

IoT – Internet of Things.

JavaScript - programming language that conforms to the ECMAScript specification.

JSCH library - pure Java implementation of SSH2.

LAMP - an acronym of the names of the original four open-source components: Linux, Apache, MYSQL, and PHP stack.

LAN – Local Area Network.

LED – Light Emitting Diode.

Linux - a family of open source Unix-like operating systems based on the Linux kernel.

MAIDS - Meis Alarm Intrusion Detection System.

MariaDB – Maria database similar to MySQL.

MySQL - open-source relational database management system.

NAT - Network address translation is a method of remapping one IP address space into another.

NOOBS - an easy operating system installer which contains Raspbian and LibreELEC.

NPTH - Non Plated Through Hole.

OACETT - Ontario Association of Certified Engineering Technicians and Technologists.

OFRS - Optical Fiducial Recognition Systems.

OS – Operating System.

PIR – Passive Infrared Sensor.

PCB Board - printed circuit board that mechanically supports and electrically connects electrical or electronic components.

PHP - general-purpose scripting language that is especially suited to web development.

phpMyAdmin - free software tool written in PHP, intended to handle the administration of MySQL over the Web.

Pixel - physical point in a raster image, or the smallest addressable element in an all points addressable display.

Pushover- A service that makes it easy to get real-time notifications on your Android, iPhone, iPad, and Desktop.

Python - interpreted, high-level, general-purpose programming language.

RAM – Random Access Memory.

Raspbian – Raspberry Pi Foundation's official supported operating system.

Raspberry Pi - small single-board computer.

RDBS - relational database system.

RoHS - Restriction of Hazardous Substances directive.

RPi.GPIO library – Package that provides a class to control the GPIO on a Raspberry Pi.

SDHC - Secure Digital High Capacity format.

SMS – Short Message Service is a text messaging service component of most telephone, Internet, and mobile device systems.

Solderless – does not require soldering to make connections.

SPST - ingle Pole Single Throw switch.

Toast Message – a small, non-disruptive popup for success or

information messages that disappears automatically after a few seconds.

USB – Universal Serial Bus.

Wi-Fi – a wireless networking technology that uses radio waves to provide wireless high-speed Internet and network connections.

Wireless - computer network where there is no physical wired connection between sender and receiver, but rather the network is connected by radio waves and/or microwaves to maintain communications.

WebView - objects that allows the display of web content as part of the activity layout.

WPA2 Authentication - Short for Wi-Fi Protected Access 2 - Pre-Shared Key, and also called WPA or WPA2 Personal, it is a method of securing your network using WPA2 with the use of the optional Pre-Shared Key (PSK) authentication, which was designed for home users without an enterprise authentication server.

XAMPP – Linux, Apache, MySQL, PHP and Pearl stack.

XML - Extensible Markup Language.

Appendix A: Installing OS Image to SCHD Card

This appendix explains how to install a Raspberry Pi operating system image on an SD card that will be used in The Raspberry PI 4 platform. It is recommended that new users download NOOBS because it is designed to be very easy to use.

Step 1: Download the Raspbian OS image

Official images for the Raspbian operating systems are available to download from the Raspberry Pi website Downloads page at https://www.raspberrypi.org/downloads/. If decompression is required, it is suggested that the following software packages be used according to its compression:

1. 7-Zip or WinRAR (Windows)
2. The Unarchiver (Mac)
3. Unzip (Linux)

Step 2: Download balenaEtcher Software

The balenaEtcher utility is a free and open-source utility used for writing image files such as .iso and .img files, as well as zipped folders onto storage media to create live SD cards and USB flash drives. For the Windows operating system, balenaEtcher can be downloaded from: https://www.balena.io/etcher/.

Step 3: Writing an image to the SD card

It is important to note that before writing to the SD card, the SD card requirements are checked. To write your image with balenaEtcher:

1. Connect an SD card reader with the SD card inside.

2. Open balenaEtcher and select from your hard drive the Raspberry Pi .img or .zip file you wish to write to the SD card.
3. Select the SD card you wish to write your image to.
4. Review your selections and click 'Flash!' to begin writing data to the SD card.

Note: for Linux users, zenity might need to be installed on your machine for balenaEtcher to be able to write the image on your SD card.

Appendix B: MAIDS Final Python Source Code

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

#========================================================================
=========
# BUILT-IN/GENERIC IMPORTS FOR MAIDS
#========================================================================
=========
import RPi.GPIO as GPIO
import time
import datetime as datetime
import smtplib
import ssl
import os
import http.client
import urllib
import vlc
import mysql.connector

#========================================================================
=========
# MODULES IMPORTED
#========================================================================
=========
from email.mime.multipart import MIMEMultipart
from email.mime.base import MIMEBase
from email.mime.text import MIMEText
from email.utils import formatdate
from email import encoders
from subprocess import call
from mysql.connector import Error

#=================================================================
=========
#************************ GLOBALS *****************************************
#=================================================================
=========
#
# MAIDS PROJECT-SPECIFIC GLOBALS ******************************************
#
__author__ = "Claudio F. Meis"
__projectname__ = "MAIDS PROJECT"
__date__ = "October 12, 2019"
__copyright__ = "Copyright (c) 2019, Claudio F. Meis, The MAIDS project"
__credits__ = ["Claudio F. Meis","Conor Meis"]
__license__ = "MIT License"
__version__ = "1.0.0"
__maintainer__ = "Claudio F. Meis"
__email__ = "cfpm@live.ca,claudiomeis57@gmail.com"
__status__ = "Dev"
__school__ = "Humber College"
__program__ = "Computer Engineering Technology"
__course__ = "CENG317"
__studentno__ = "N00674230"
__instructor__ = "Mr. Austin Tian"
__description__ = "MEIS ALARM INTRUSION DETECTION SYSTEM"
#
# MAIDS PROJECT PINOUT GLOBALS ********************************************
#
__SWITCH_PIN__ = 7
```

```
__GREEN_LED_PIN__ = 11
__RED_LED_PIN__ = 13
__MOTION_PIN__ = 29
__SOUND_PIN__ = 31
__BUZZER_PIN__ = 33
chan_list2 = (__RED_LED_PIN__,__GREEN_LED_PIN__,__BUZZER_PIN__)
#
# MAIDS PROJECT SOUND ACTIVATION GLOBALS **********************************
#
__sounds__ = ( '/home/pi/mp3/leaveroom.mp3', '/home/pi/mp3/10.mp3',
'/home/pi/mp3/9.mp3', '/home/pi/mp3/8.mp3', \
                '/home/pi/mp3/7.mp3', '/home/pi/mp3/6.mp3', '/home/pi/mp3/5.mp3',
'/home/pi/mp3/4.mp3', \
                '/home/pi/mp3/3.mp3', '/home/pi/mp3/2.mp3', '/home/pi/mp3/1.mp3',
'/home/pi/mp3/maids_activated.mp3' )

__sounds2__ = ( '/home/pi/mp3/s2.wav', '/home/pi/mp3/51267034.mp3')

__sounds3__ =  ( '/home/pi/mp3/deactivating.mp3', '/home/pi/mp3/shutdown.mp3',
'/home/pi/mp3/goodbye.mp3')

__count__ = 11
#
#MySQL DATABASE GLOBALS ***************************************************
#
__address__ = "1234 Brook Road, Etobicoke, ON."
__location__ = "Room"
__reportingperson__ = "CONOR PATRICK JOSEPH MEIS"
__contactphone__ = "647-123-4567"
__contactemail__ = "cfpm@live.ca"
intrusiondate = datetime.datetime.now()
```

```
#=============================================================
=========
#************************ END OF GLOBALS **********************************
#=============================================================
=========

#=============================================================
=========
# MAIDS PYTHON SCRIPT DESCRIPTION
#=============================================================
=========
#
# This script will set up the inputs/oputput pins for the Meis
# Alarm Intrusion Detection System (MAIDS) and manipulated them
# to provide a fully functional intrusion detection system based
# on motion and sound sensors, warning lights and buzzer alarm.
# Email notifications with pictures, Android Push Notifications,
# SMS message and phone call are sent upon intrusion being
#   registered by sensors.

#=============================================================
=========
# MAIDS MIT LICENSE INFORMATION
#=============================================================
=========
#
# Copyright 2019 Claudio F. Meis - MAIDS PROJECT
#
# Permission is hereby granted, free of charge, to any person obtaining a
# copy of this software and associated documentation files (the "Software"),
# to deal in the Software without restriction, including without limitation
# the rights to use, copy, modify, merge, publish, distribute, sublicense,
# and/or sell copies of the Software, and to permit persons to whom the
# Software is furnished to do so, subject to the following conditions:
#
# The above copyright notice and this permission notice shall be included
# in all copies or substantial portions of the Software.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS
# OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
# FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL
# THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
# LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
# OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
# THE SOFTWARE.

#************************************************************************
# START OF FUNCTIONS BLOCK
#************************************************************************
# Function block contains the following functions:
# 1. set_up_gpio()          --> line 160
# 2. maids()                --> line 184
# 3. notify()               --> line 253
# 4. alarm()                --> line 275
# 5. alert()                --> line 291
# 6. sound(sound)            --> line 302
# 7. motion(motion)          --> line 321
# 8. send_mail()             --> line 340
# 9. send_androidpush()      --> line 382
# 10. appendtodb()           --> line 403
# 11. intruderwarning()      --> line 424
# 12. siren()                --> line 437
# 13. activationwarning()    --> line 450
# 14. dbinsert()             --> line 466

#==================================================================
=========
# 1. set_up_gpio() --> SETUP GPIO NUMBERING SYSTEM AND I/O's
#==================================================================
=========
def set_up_gpio():

    GPIO.setmode(GPIO.BOARD)       # SETUP NUMBERING SYSTEM FOR GPIO
    GPIO.setwarnings(False)        # DISABLE WARNINGS

    # SETTING UP INPUT/OUTPUT PINS ******************************************
    GPIO.setup(__SWITCH_PIN__, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
#SWITCH - IN
    GPIO.setup(__MOTION_PIN__, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
#MOTION SENSOR - PIR - IN
    GPIO.setup(__SOUND_PIN__, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
#SOUND SENSOR - IN

    GPIO.setup(__RED_LED_PIN__, GPIO.OUT)
    GPIO.setup(__GREEN_LED_PIN__, GPIO.OUT)
    GPIO.setup(__BUZZER_PIN__, GPIO.OUT)
    GPIO.output(chan_list2,(GPIO.LOW,GPIO.HIGH,GPIO.HIGH))   #RED LED
OFF/GREEN LED OFF/BUZZER OFF
    time.sleep(1)


#***************************************************************************

#===========================================================================
=========
# 2. maids() --> WELCOME SCREEN FUNCTION FOR MAIDS
#===========================================================================
=========
def maids():
    '''
    MAIDS Function -- Displays MAIDS onscreen logo and project information.
    Parameters:
    :param none:
    Input Data:
    See MAIDS PROJECT SPECIFIC INFORMATION section

    '''
    os.system("clear")
    print("                          \\\\\V/// ")
    print("                       \V/\/\\\\\V// ")
    print("                     \\\`      \\\\\V//")
    print("                    \\       ||\      \ ")
    print("                    \  \\   //     _\  `\ ")
    print("                   /  /. \  \\    /O.    `\,")
    print("                  //  |__\\ //\         . __\ ")
    print("                /`           //\\      , .\ / ")
print("            \\\\        //\       ___|")
print("           ///\\          \\    `   \ ")
print("         //////////\\\\      //__       |")
print("         |`  \\V/////\\       \_ \_______|")
print("        |     \\\V/\V////\\\   \ ")
print("       ./      \\\V///////\\     |\ ")
print("       |        \\\V///\V/\\\\\\\\ ")
print("       |          \\V//      \\\\\ ")
print("       |          \\V/         V/ ")
print("       |            V         \ |")
print("       |            `          \|")
print("       | |                  \                    / ")
print("        | |          \          \                 // ")
print("        | |                      \               //// ")
print("        | |            .          `|             ///// ")
print("        | |                        `\            \V/// ")
print("         \`|                         `|           \\||/ ")
print("           | |           \             `|  ,--.        \ \,")
print("           |  \                         |./    `\       | |")
print("            |  |                                |       | |")
print("            |___|           .                    |       | |")
print("            /   |                               |       | |")
print("            |    |                              ;       | |")
print("            |                                   |       | |")
print("          __|                                    /`      /` ;")
print("        /   \                       ,       ; \     ,` ,/ ")
print("        \_____\           \       \,/_______________|___.' ,`")
print(" ")
print("     MMM  MM  MMM   AAAAAAAA   IIIIIIII   DDDDDDD    SSSSSSSS")
print("     MMMM MM MMMM   AAAAAAAA     IIII     DDDDDDD    SSSSSSSS")
print("     MMMM MM MMMM   AAA  AAA     IIII    DDD  DDD   SSS     ")
```

```
   print("     MMMM MM MMMM   AAA  AAA    IIII    DDD  DDD   SSS    ")
   print("     MMMMMMMMMMMMM   AAAAAAAA    IIII    DDD  DDD   SSSSSSSS")
   print("     MMM MMMM MMM   AAAAAAAA    IIII    DDD  DDD       SSS")
   print("     MMM  MM  MMM   AAA  AAA    IIII    DDD  DDD       SSS")
   print("     MMM  MM  MMM   AAA  AAA    IIII    DDDDDDD   SSSSSSSS")
   print("     MMM  MM  MMM   AAA  AAA  IIIIIII  DDDDDDD   SSSSSSSS")
   print(" ")
   print("
==========================================================================
=========================\n")
   print("     Project Name: " + __projectname__ + " - Description: " + __description__)
   print("     Author: " + __author__ + " - Stud. No: " + __studentno__ + " Email: " +
__email__)
   print("     Date: " + __date__ + " - " + __copyright__)
   print("     License: " + __license__ + " - Version: " + __version__ + " - Maintainer: " +
__maintainer__)
   print("     School: " + __school__ + " - Program: " + __program__)
   print("     Program: " + __program__)
   print("     Course: " + __course__ + " - Instructor: " + __instructor__)
   print("
==========================================================================
=========================\n")
   time.sleep(2)
#**************************************************************************


#=========================================================================
=========
# 3. notify() --> UPDATE DATABASE/SEND EMAIL/SEND PUSH TO ANDROID
PHONE FUNCTION
#=========================================================================
=========
def notify():
   '''
   MAIDS Function -- Appends text database/send email/sends android push
notification
   sends SMS and calls Android phone.
   Parameters:
   :param none:
   Input Data:
   none
   '''
   #SENDING PUSH NOTIFICATION TO ANDROID PHONE
   print("Sending push notification to Android phone...")
   send_androidpush()
   #SENDING SMSNOTIFICATION TO ANDROID PHONE
   print("Sending SMS notification to Android phone...")
   sendsms()
   #TAKING PIC AND SENDING EMAIL
   print("Sending Intrusion Alert to email...")
   send_email()
   #APPEND TEXT DATABASE/SENDG EMAIL WITH PICTURE/SEND ANDROID
PUSH NOTIFICATION - INTRUSION ALERT
   print("Appending Intrusion Alert to Database...")
   appendtodb()
   #CALLING ANDROID PHONE
   print("Calling Android phone...")
   callphone()
   time.sleep(1)
#*************************************************************************

#================================================================
=========
# 4. alarm() --> SET LIGHT/SOUND ALARM FORMAT
#=========================================================================
=========
def alarm():
   '''
   MAIDS Function -- Sets LED flashing/sound pattern..
   Parameters:
   :param none:
   Input Data:
   none
   '''
   GPIO.output(chan_list2,(GPIO.HIGH,GPIO.LOW,GPIO.LOW))   #RED LED/BUZZER
ON & GREEN LED OFF
   time.sleep(0.2)

   GPIO.output(chan_list2,(GPIO.LOW,GPIO.HIGH,GPIO.HIGH))   #RED LEDBUZZER
OFF & GREEN LED ON
   time.sleep(0.2)
#******************************************************************************

#=========================================================================
=========
# 5. alert() --> SOUND SENSOR FUNCTION
#=========================================================================
=========
def alert():
   for x in range(1, 6):
       alarm()
   GPIO.output(chan_list2,(GPIO.LOW,GPIO.LOW,GPIO.HIGH))   #RED LED/GREEN
LED/BUZZER OFF
   time.sleep(0.2)
   intruderwarning()
   notify()
   siren()
   time.sleep(1)

#=================================================================
=========
# 6. sound(sound) --> SOUND SENSOR FUNCTION
#=================================================================
=========
def sound(sound):
   ''' MAIDS Function -- Detects sound.
   Parameters:
   :param none:
   Input Data:
   none
   '''
   GPIO.output(__GREEN_LED_PIN__,GPIO.HIGH)
   if GPIO.input(sound) == False:
       print("MAIDS - MAIDS ALERT - Sound Detected!")
       player = vlc.MediaPlayer('/home/pi/mp3/sounddetected.mp3')
       player.play()
       time.sleep(2)
       alert()
       GPIO.output(__GREEN_LED_PIN__, GPIO.HIGH)
   else:
       GPIO.output(__GREEN_LED_PIN__,GPIO.HIGH)
#**************************************************************************

#=================================================================
=========
# 7. motion(motion) --> MOTION SENSOR FUNCTION
#==================================================================
=========
def motion(motion):
   ''' MAIDS Function -- Detects motion.
   Parameters:
   :param none:
   Input Data:
   none
   '''
   GPIO.output(__GREEN_LED_PIN__,GPIO.HIGH)
   if GPIO.input(motion) == False:
       print("MAIDS ALERT - Motion Detected!")
       player = vlc.MediaPlayer('/home/pi/mp3/motiondetected.mp3')
       player.play()
       time.sleep(1)
       alert()
       GPIO.output(__GREEN_LED_PIN__, GPIO.HIGH)
   else:
       GPIO.output(__GREEN_LED_PIN__,GPIO.HIGH)
#**************************************************************************

#==================================================================
=========
# 8. send_email() --> SENDING ALERT WITH PICTURE TO RECIPIENT
#==================================================================
=========
def send_email():
   ''' MAIDS Function -- Sends picture of intruder to email address.
   Parameters:
   :param none:
```

```
    Input Data:
    none

    '''
    os.system("fswebcam -r 1280x720 --title 'MAIDS INTRUSION ALERT' --subtitle '1234 BROOK ROAD, ETOBICOKE, ON.' --timestamp '%Y-%m-%d %H:%M (%Z)' --info 'LIVING ROOM ENTRY' --jpeg -1 --save /home/pi/webcam/image.jpg")
    recipient = "cfpm@live.ca"                    # recipient email
    sender = "claudiomeis57@gmail.com"            # sender email
    subject = "INTRUSION DETECTED BY MAIDS!\n1234 BROOK ROAD, ETOBICOKE, ON. - LIVING ROOM ENTRY\n"  # email Subject
    msg = MIMEMultipart()
    msg['Subject'] = subject
    msg['From'] = sender
    msg['To'] = recipient
    msg.preamble = "INTRUSION DETECTED BY MAIDS!\n1234 BROOK ROAD, ETOBICOKE, ON. - LIVING ROOM ENTRY\n"
    part = MIMEBase('application', "octet-stream")
    part.set_payload(open("/home/pi/webcam/image.jpg", "rb").read())
    encoders.encode_base64(part)
    part.add_header('Content-Disposition', 'attachment; filename="/home/pi/webcam/image.jpg"')    # File/format name
    msg.attach(part)

    try:
      # INFORMATION FOR SMTP SERVER
      port = 587                              # port For starttls
      smtp_server = "smtp.gmail.com"          # server address
      s = smtplib.SMTP(smtp_server,port)      # Setup SMTP server for Gmail
      s.starttls()
      # GMAIL LOGIN INFORMATION
```

```
        username = "claudiomeis57@gmail.com"       # Setup username for login
        password = "Srgawain2264"                  # Setup password for login
        s.login(username,password)                 # gmail.com login username/password
        # SENDING EMAIL TO RECIPIENT
        s.sendmail(sender, recipient, msg.as_string())    # send email
        s.quit()
    except:
        print ("Error: Unable to send email")          # Exception
#*************************************************************************

#================================================================
=========
# 9. send_androidpush() --> SETUP ANDROID PUSH NOTIFICATION INFORMATION
WITH PUSHOVER SERVICE
#================================================================
=========
def send_androidpush():
    ''' MAIDS Function --Sends push notification to Android phone.
    Parameters:
    :param none:
    Input Data:
    token id: "ahztmyszcui2w1svm21bbo813yie44"
    user id: "ujt5f9osjotntdhapm64ab4dg8jt2m"
    '''
    conn = http.client.HTTPSConnection("api.pushover.net:443")
    conn.request("POST", "/1/messages.json",
    urllib.parse.urlencode({
        "token": "ahztmyszcui2w1svm21bbo813yie44",
        "user": "ujt5f9osjotntdhapm64ab4dg8jt2m",
        "message": "INTRUSION DETECTED BY MAIDS - ALERT SENT! - REPORTING
PERSON: JOHN SMITH CONTACT: 647-123-4567 EMAIL: cfpm@live.ca -
INTRUSION ADDRESS: 1234 BROOK ROAD, ETOBICOKE, ON. LOCATION: LIVING
ROOM",
    }), { "Content-type": "application/x-www-form-urlencoded" })
    conn.getresponse()
#**********************************************************************

#=================================================================
=========
# 10. appendtodb() --> APPEND INTRUSION DATA TO TEXT and MySQL DATABASE
#=================================================================
=========
def appendtodb():
    ''' MAIDS Function -- Updates text database.
    Parameters:
    :param none:
    Input Data:
    none
    '''
    # APPENDING INTRUSION INFORMATION TO TEXT DATABASE
    filename = "/home/pi/maids_alarm_record.txt"                  #assign text file to update
    now = datetime.datetime.now().strftime("%Y-%m-%d_%H.%M.%S")    #create
timestamp
    file= open(filename,"a+")                                 #open file to append
    file.write("\n")                                   #write info to file
    #file.write(f"INTRUSION DETECTED -- ALARM ACTIVATED! EMAIL WITH PHOTO
SENT! DATE/TIME: {now}\n")
    file.write("REPORTING PERSON: JOHN SMITH CONTACT: 647-123-4567 EMAIL:
cfpm@live.ca\n")
    file.write("INTRUSION ADDRESS: 1234 BROOK ROAD, ETOBICOKE, ON.
LOCATION: LIVING ROOM\n\n")
    file.close()                                       #close file
    # APPENDING INTRUSION INFORMATION TO MySQL DATABASE
    dbinsert(__address__, __location__, intrusiondate, __reportingperson__,
__contactphone__,__contactemail__)
#***************************************************************************

#==================================================================
=========
# 11. intruderwarning() -- > PLAY INTRUDER WARNING MESSAGE
#==================================================================
=========
def intruderwarning():
    ''' MAIDS Function -- Plays warning message to intruder through vlc.
    Parameters:
    :param none:
    Input Data: /home/pi/mp3/s2.mp3 file
    '''
    player = vlc.MediaPlayer(__sounds2__[1])
    player.play()
    time.sleep(1)
#***************************************************************************

#==================================================================
=========
# 12. siren() -- > PLAY SIREN ALARM SOUND
#==================================================================
=========
def siren():
    ''' MAIDS Function -- Plays a siren alarm sound through vlc.
    Parameters:
    :param none:
```

```
    Input Data: /home/pi/mp3/51267034.mp3 file
    '''
    player = vlc.MediaPlayer(__sounds2__[0])
    player.play()
    time.sleep(0.20)
#**************************************************************************

#==================================================================
=========
# 13. activationwarning() -- > PLAY SIREN ALARM SOUND
#==================================================================
=========
def activationwarning():
    ''' MAIDS Function -- Plays a siren alarm sound through playsound module.
    Parameters:
    :param none:
    Input Data: __sounds__ list
    :Needed: pip install playsound
    '''
    for index in range(len(__sounds__)):
        player = vlc.MediaPlayer(__sounds__[index])
        player.play()
        time.sleep(3)
#**************************************************************************

#==================================================================
=========
# 14. dbinsert() -- > INSERT INTRUSION INFORMATION INTO MYSQL DATABASE
#==================================================================
=========
def dbinsert(address, location, intrusiondate, reportingperson, contactphone,
contactemail):
    ''' MAIDS Function -- inserts intrusion particulars to MySQl database.
    Parameters:
    :param address:
    :param location:
    :param intrusiondate:
    :param reportingperson:
    :param contactphone:
    :param contactemail:
    :Input Data: __sounds__ list
    '''
    try:
        connection = mysql.connector.connect(host='localhost',
                                             database='maids1',
                                             user='root',
                                             password='')
        cursor = connection.cursor()
        mySql_insert_query = """INSERT INTO maidsintrusion (address, location,
intrusiondate, reportingperson, contactphone, contactemail)  \
                                VALUES (%s, %s, %s, %s, %s, %s) """
        a = (address, location, intrusiondate, reportingperson, contactphone, contactemail)
        cursor.execute(mySql_insert_query, a)
        connection.commit()
        print("Record inserted successfully into MAIDS-DB")
    except mysql.connector.Error as error:
        print("Failed to insert into MySQL table {}".format(error))
    finally:
        if (connection.is_connected()):
            cursor.close()
            connection.close()
            print("MySQL connection - CLOSED")

# 15. shutdown() -- > INSERT INTRUSION INFORMATION INTO MYSQL DATABASE
#===============================================================
=========
def shutdown():
    ''' MAIDS Function -- shutdown MAIDS.
    Parameters:
    none
    :Input Data: mp3 sounds
    '''
    for index in range(len(__sounds3__)):
        player = vlc.MediaPlayer(__sounds3__[index])
        player.play()
        time.sleep(3)

#===============================================================
=========
# 16. sendsms() -- > SEND SMS INTRUSION ALERT
#===============================================================
=========
def sendsms():
    # Download the helper library from https://www.twilio.com/docs/python/install
    from twilio.rest import Client

    # Your Account Sid and Auth Token from twilio.com/console
    # DANGER! This is insecure. See http://twil.io/secure
    account_sid = 'AC7b062601eca1c3d78642021e82e0eb7b'
    auth_token = 'f83955e36e1df48109a15369eb73fc2e'
    client = Client(account_sid, auth_token)
    message = client.messages \
                .create(
                    body="MAIDS - INTRUSION ALERT - 1234 Brook Road, Etobicoke, ON. -
Living Room Intrusion Detected!",
                    from_='+16475592395',
                    to='+14372304874'
                 )

    print("Mesaage sent ID: " + message.sid)

#===================================================================
=========
# 17. callphone() -- > CALL PHONE WITH INTRUSION ALERT
#===================================================================
=========
def callphone():
    # Download the helper library from https://www.twilio.com/docs/python/install
    from twilio.rest import Client

    # Your Account Sid and Auth Token from twilio.com/console
    # DANGER! This is insecure. See http://twil.io/secure
    account_sid = 'AC7b062601eca1c3d78642021e82e0eb7b'
    auth_token = 'f83955e36e1df48109a15369eb73fc2e'
    client = Client(account_sid, auth_token)

    call = client.calls.create(
        url='http://demo.twilio.com/docs/voice.xml',
        from_='+16475592395',
        to='+14372304874'
    )
    print("Phone call ID: " + call.sid)


#***************************************************************************
# END OF FUNCTIONS SECTION
#***************************************************************************

#==================================================================
=========
# STARTTING MAIN SCRIPT
#==================================================================
=========
if __name__ == '__main__':
    #Setup GPIO Information
    set_up_gpio()
    #Display MAIDS logo/project information
    maids()
    print("MAIDS Surveillance Mode - Arming...")
    #warning to vacate the room
    activationwarning()
    print("MAIDS Surveillance Mode - Armed and Active...\n")
    # Sound sensor: Detect when pin goes HIGH or LOW
    GPIO.add_event_detect(__SOUND_PIN__, GPIO.BOTH, bouncetime=200)
    # assign function to PIN & run it
    GPIO.add_event_callback(__SOUND_PIN__, sound)
    # Motion sensor: Detect when pin goes HIGH or LOW
    GPIO.add_event_detect(__MOTION_PIN__, GPIO.BOTH, bouncetime=200)
    # assign function to PIN, and run it
    GPIO.add_event_callback(__MOTION_PIN__, motion)

    try:
```

```
    while True:         # infinite loop
        time.sleep(1)

except KeyboardInterrupt:
    shutdown()
    #RED LED OFF/GREEN LED ON
    GPIO.output(chan_list2,(GPIO.LOW,GPIO.LOW,GPIO.HIGH))
    time.sleep(1)
    GPIO.cleanup()        #cleanup

GPIO.cleanup()            #cleanup
```

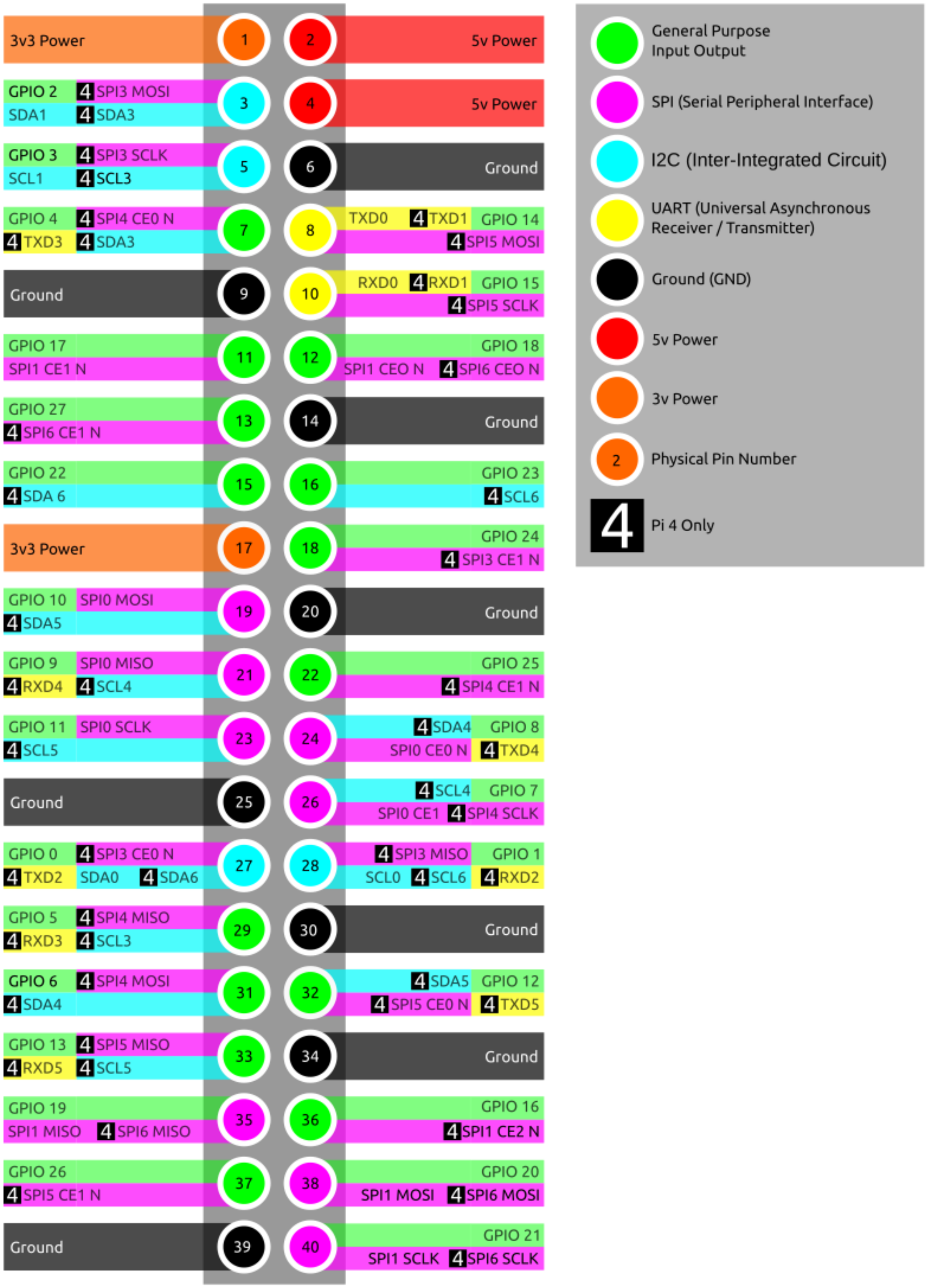
}

Appendix C: MAIDS Wireless Connection Configuration

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=CA

network={
        ssid="Meis908070-5G"
        psk="Srgawain2264"
        disabled=1
}

network={
        ssid="FkeV-Y2ZwbQ"
        psk="cmeis2264"
        disabled=1
}


network={
        ssid="myWi-Fi@Humber"
        key_mgmt=WPA-EAP
        auth_alg=OPEN
        eap=PEAP
        identity="mscl0015"
        password="Srgawain2264"
        phase1="peaplabel=0"
        phase2="auth=MSCHAPV2"
        priority=999
        proactive_key_caching=1
}
```

```
network={
	ssid="eduroam"
	key_mgmt=WPA-EAP
	auth_alg=OPEN
	eap=PEAP
	identity="mscl0015@humber.ca"
	password="Srgawain2264"
	phase2="auth=MSCHAPV2"
	priority=999
	proactive_key_caching=1
	disabled=1
}

network={
	ssid="Welcome to Humber"
	key_mgmt=NONE}
```

Appendix D: MAIDS Raspberry Pi 4 GPIO Pinout Reference



Left	Pin	Pin	Right
3v3 Power	1	2	5v Power
GPIO 2 · 4 SPI3 MOSI SDA1 · 4 SDA3	3	4	5v Power
GPIO 3 · 4 SPI3 SCLK SCL1 · 4 SCL3	5	6	Ground
GPIO 4 · 4 SPI4 CE0 N 4 TXD3 · 4 SDA3	7	8	TXD0 · 4 TXD1 · GPIO 14 4 SPI5 MOSI
Ground	9	10	RXD0 · 4 RXD1 · GPIO 15 4 SPI5 SCLK
GPIO 17 SPI1 CE1 N	11	12	GPIO 18 SPI1 CEO N · 4 SPI6 CEO N
GPIO 27 4 SPI6 CE1 N	13	14	Ground
GPIO 22 4 SDA 6	15	16	GPIO 23 4 SCL6
3v3 Power	17	18	GPIO 24 4 SPI3 CE1 N
GPIO 10 · SPI0 MOSI 4 SDA5	19	20	Ground
GPIO 9 · SPI0 MISO 4 RXD4 · 4 SCL4	21	22	GPIO 25 4 SPI4 CE1 N
GPIO 11 · SPI0 SCLK 4 SCL5	23	24	4 SDA4 · GPIO 8 SPI0 CE0 N · 4 TXD4
Ground	25	26	4 SCL4 · GPIO 7 SPI0 CE1 · 4 SPI4 SCLK
GPIO 0 · 4 SPI3 CE0 N 4 TXD2 · SDA0 · 4 SDA6	27	28	4 SPI3 MISO · GPIO 1 SCL0 · 4 SCL6 · 4 RXD2
GPIO 5 · 4 SPI4 MISO 4 RXD3 · 4 SCL3	29	30	Ground
GPIO 6 · 4 SPI4 MOSI 4 SDA4	31	32	4 SDA5 · GPIO 12 4 SPI5 CE0 N · 4 TXD5
GPIO 13 · 4 SPI5 MISO 4 RXD5 · 4 SCL5	33	34	Ground
GPIO 19 SPI1 MISO · 4 SPI6 MISO	35	36	GPIO 16 4 SPI1 CE2 N
GPIO 26 4 SPI5 CE1 N	37	38	GPIO 20 SPI1 MOSI · 4 SPI6 MOSI
Ground	39	40	GPIO 21 SPI1 SCLK · 4 SPI6 SCLK

🟢	General Purpose Input Output
🟣	SPI (Serial Peripheral Interface)
🔵	I2C (Inter-Integrated Circuit)
🟡	UART (Universal Asynchronous Receiver / Transmitter)
⚫	Ground (GND)
🔴	5v Power
🟠	3v Power
2	Physical Pin Number
4	Pi 4 Only

Appendix E: MySQL Configuration

```
--
phpMyAdmi
n SQL
Dump
            -- version 4.9.1
            -- https://www.phpmyadmin.net/
            --
            -- Host: 127.0.0.1
            -- Generation Time: Oct 29, 2019 at 02:42 AM
            -- Server version: 10.4.8-MariaDB
            -- PHP Version: 7.3.10
            SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
            SET AUTOCOMMIT = 0;
            START TRANSACTION;
            SET time_zone = "+00:00";
            /*!40101 SET
            @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIE
            NT */;
            /*!40101 SET
            @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RE
            SULTS */;
            /*!40101 SET
            @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTI
            ON */;
            /*!40101 SET NAMES utf8mb4 */;
            --
            -- Database: `maids1`
            --
            -- --------------------------------------------------------
```

```
--
-- Table structure for table `maidsintrusion`
--
CREATE TABLE `maidsintrusion` (
  `id` int() NOT NULL AUTO_INCREMENT PRIMARY,
  `address` varchar(25) NOT NULL,
  `location` varchar(25) NOT NULL,
  `intrusiondate` varchar(30) NOT NULL,
  `reportingperson` varchar(25) NOT NULL,
  `contactphone` varchar(25) NOT NULL,
  `contactemail` varchar(25) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
--
-- Indexes for dumped tables
--
--
-- Indexes for table `maidsintrusion`
--
ALTER TABLE `maidsintrusion`
  ADD PRIMARY KEY (`id`);
--
-- AUTO_INCREMENT for dumped tables
--
--
-- AUTO_INCREMENT for table `maidsintrusion`
--
ALTER TABLE `maidsintrusion`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
COMMIT;
/*!40101 SET
CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
```

```
/*!40101 SET
CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESUL
TS */;
/*!40101 SET
COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION
*/;
```

Appendix F: MAIDS Comparison with Products in Market

The top five home/business security system in Canada (Privacy Canada, 2020) similar to MEIS-Alert are:

Provider	Monitoring Price (Monthly)	Installation Type	Smart Home Integrations
Vivint	$29.99–$44.99	Professional	Top-notch equipment and detectors • Home automation integrations • Mobile control • Pricey equipment • Camera support only on $44.99 per month plan
ADT /mo. Yes •	$33.99–$53.99	Professional	Reliable monitoring • Wide availability • Pricey monitoring plans • No camera support • Mobile control on premium plan
Frontpoint	$34.99–$49.99	DIY	Easy installation process • Excellent customer support • Low equipment cost • 3-year contract required • Camera and phone access only on Ultimate Plan
Brinks	$29–$39	DIY	Multi-channel security alerts • Fast response times • 3-year contract required • Camera and phone access only on Home Complete with Video Plan

Ring Alarm	$5–$15	DIY	Affordable equipment • Low-priced professional monitoring • Camera and phone access on all plans • No Google Assistant support 5] – [7] Multiple references.

Appendix G: Experts in Marketing to Asian Markets

1. AAAZA, Inc.
2. DAE
3. Emerging Networks, LLC
4. Ethnic Technologies
5. INQUIRER.net
6. Interlex Communications, Inc.
7. Interviewing Service of America (ISA)
8. MediaMorphosis
9. Multicultural Marketing Resources, Inc.
10. Muse Communications, Inc.
11. NDTV
12. Opinion Access Corp.
13. Paragon Language Services, Inc.
14. T.D. Wang Advertising Group, LLC
15. UWG
16. VanguardComm

Appendix H: Dual Color Led Datasheet

Dual-Color LED Introduction A dual-color light emitting diode (LED) is capable of emitting two different colors of light, typically red and green, rather than only one color. It is housed in a 3mm or 5mm epoxy package. It has 3 leads; common cathode or common anode is available. A dual-color LED features two LED terminals, or pins, arranged in the circuit in anti-parallel and connected by a cathode/anode. Positive voltage can be directed towards one of the LED terminals, causing that terminal to emit light of the corresponding color; when the direction of the voltage is reversed, the light of the other color is emitted. In a dual-color LED, only one of the pins can receive voltage at a time. As a result, this type of LED frequently functions as indicator lights for a variety of devices, including televisions, digital cameras, and remote controls.

Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 1 * Network cable (or USB wireless network adapter)
- 1 * Dual-color LED module
- 1 * 3-pin anti-reverse cable

Experimental Principle Connect pin R and G to GPIOs of Raspberry Pi, change the color of the LED from red to green by programming, and then use PWM to make it flash various mixed colors. The schematic diagram:

Experimental Procedures Step 1: Build the circuit

Raspberry Pi Dual-Color LED Module GPIO0 R GND GND GPIO1 G

For C language users: Step 2: Change directory cd /home/pi/SunFounder_SensorKit_for_RPi2/C/01_dule_color_led/ Step 3: Compile gcc dule_color_led.c –lwiringPi -lpthread Step 4: Run sudo ./a.out

For Python users: Step 2: Change directory cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/ Step 3: Run sudo python 01_dule_color_led.py

Now you can see the dual-color LED changes from red to green alternately, as well as flashing a mixed color during the alternation.

C Code

```
#include <wiringPi.h>
#include <softPwm.h>
#include <stdio.h>
#define uchar unsigned char
#define LedPinRed 0 #define LedPinGreen 1
void ledInit(void) { softPwmCreate(LedPinRed, 0, 100); softPwmCreate(LedPinGreen,0,
100); }
void ledColorSet(uchar r_val, uchar g_val) { softPwmWrite(LedPinRed, r_val);
softPwmWrite(LedPinGreen, g_val); }
int main(void) { int i;
if(wiringPiSetup() == -1){ //when initialize wiring failed,print messageto screen
        printf("setup wiringPi failed !");
        return 1;
}
//printf("linker LedPin : GPIO %d(wiringPi pin)\n",LedPin); //when initialize wiring
successfully,print message to screen

ledInit();

while(1){
        ledColorSet(0xff,0x00);   //red
        delay(500);
        ledColorSet(0x00,0xff);   //green
        delay(500);
        ledColorSet(0xff,0x45);
        delay(500);
        ledColorSet(0xff,0xff);
        delay(500);
        ledColorSet(0x7c,0xfc);
        delay(500);
}

return 0;
}
```

Python Code

```
#!/usr/bin/env python
import RPi.GPIO as GPIO
import time
colors = [0xFF00, 0x00FF, 0x0FF0, 0xF00F] pins = (11, 12) # pins is a dict
GPIO.setmode(GPIO.BOARD) # Numbers GPIOs by physical location GPIO.setup(pins,
GPIO.OUT) # Set pins' mode is output GPIO.output(pins, GPIO.LOW) # Set pins to
LOW(0V) to off led
p_R = GPIO.PWM(pins[0], 2000) # set Frequece to 2KHz p_G = GPIO.PWM(pins[1],
2000)
p_R.start(0) # Initial duty Cycle = 0(leds off) p_G.start(0)
def map(x, in_min, in_max, out_min, out_max): return (x - in_min) * (out_max - out_min)
/ (in_max - in_min) + out_min
def setColor(col): # For example : col = 0x1122 R_val = col >> 8 G_val = col & 0x00FF
R_val = map(R_val, 0, 255, 0, 100)
G_val = map(G_val, 0, 255, 0, 100)

p_R.ChangeDutyCycle(R_val)     # Change duty cycle
p_G.ChangeDutyCycle(G_val)
```

```
def loop(): while True: for col in colors: setColor(col) time.sleep(0.5)
def destroy():
    p_R.stop()
    p_G.stop()
    GPIO.output(pins, GPIO.LOW) # Turn off all leds
    GPIO.cleanup()

if name == "main":
    try:
         loop()
     except
          KeyboardInterrupt: destroy()
```

Appendix II: Buzzer Datasheet

Introduction Buzzers can be categorized as active buzzers and passive ones (See the following picture).

Components

- 1 * SunFounder Uno board
- 1 * USB data cable
- 1 * Buzzer module
- Several jumper wires

Experimental Principle Place the pins of two buzzers face up and you can see the one with a green circuit board is a passive buzzer, while the other with a black tape, instead of a board, is an active buzzer, as shown below.

An active buzzer has a built-in oscillating source, so it will make sounds when electrified. But a passive buzzer does not have such source, so it will not beep if DC signals are used; instead, you need to use square waves whose frequency is between 2K and 5K to drive it. The active buzzer is often more expensive than the passive one because of multiple built-in oscillating circuits.

Experimental Procedures Passive Buzzer

Step 1: Build the circuit Passive Buzzer Module SunFounder Uno S ------------------------------------ D11 - ------------------------------------ GND + -------------------------------------- 5V

Step 2: Program (Please refer to the example code in LEARN -> Get Tutorial on our website)

Step 3: Compile

Step 4: Upload the sketch to SunFounder Uno

Now, you can hear the passive buzzer beep for warning.

Active Buzzer

Note: The active buzzer has built-in oscillating source, so it will beep as long as it is electrified, but it can only beep with a fixed frequency.

Step 1: Build the circuit

Active Buzzer Module SunFounder Uno

S ----------------------------------- D1

- ----------------------------------- GND

+ ----------------------------------- 5V

Step 2: Program (Please refer to the example code in LEARN -> Get Tutorial on our website)

Step 3: Compile

Step 4: Upload the sketch to SunFounder Uno

Now, you can hear the active buzzer beep.

Code for Passive

// const int buzzerPin = 3;//the buzzer pin attach to int fre;//set the variable to store the frequence value // void setup() { pinMode(buzzerPin,OUTPUT); } /**/ void loop() { for(int i = 200;i <= 800;i++) //frequence loop from 200 to 800 { tone(3,i); //turn the buzzer on delay(5); //wait for 5 milliseconds } delay(4000); //wait for 4 seconds on highest frequence for(int i = 800;i >= 200;i--)//frequence loop from 800 downto 200 { tone(3,i); delay(10); } }

Code for Active

```
int buzzer = 11;//the pin of the active buzzer void setup() {
pinMode(buzzer,OUTPUT);//initialize the buzzer pin as an output } void loop() {
unsigned char i,j; while(1) { //output an frequency for(i=0;i<80;i++) {
digitalWrite(buzzer,HIGH); delay(1);//wait for 1ms digitalWrite(buzzer,LOW);
delay(1);//wait for 1ms } //output another frequency for(i=0;i<100;i++) {
digitalWrite(buzzer,HIGH); delay(2);//wait for 2ms digitalWrite(buzzer,LOW);
delay(2);//wait for 2ms } } }
```

Appendix JI: Android Application Code

Login Activity: XML and Code

```
<?xml version = "1.0" encoding = "utf-8"?>
<RelativeLayout xmlns:android = "http://schemas.android.com/apk/res/android"
    xmlns:tools = "http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height = "match_parent"
    tools:context="com.example.maids1.MainActivity">

    <TextView android:text = "MAIDS"
        android:layout_width="wrap_content"
        android:layout_height = "wrap_content"
        android:id = "@+id/textview"
        android:textSize = "75dp"
        android:layout_alignParentTop = "true"
        android:layout_centerHorizontal = "true" />

    <TextView
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:text = "Meis Intrusion Detection System Remote Control"
        android:id = "@+id/textView"
        android:layout_below = "@+id/textview"
        android:layout_centerHorizontal = "true"
        android:textSize = "35dp" />

    <EditText
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:id = "@+id/editText"
        android:hint = "Enter Name"
        android:focusable = "true"
        android:textColorHighlight = "#ff7eff15"
        android:textColorHint = "#ffff25e6"
        android:layout_marginTop = "6dp"
        android:layout_below = "@+id/imageView"
        android:layout_alignParentLeft = "true"
        android:layout_alignParentStart = "true"
        android:layout_alignParentRight = "true"
        android:layout_alignParentEnd = "true" />

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="match_parent"
        android:layout_height="314dp"
        android:layout_below="@+id/textView"
        android:layout_marginTop="4dp"
        android:src="@drawable/a22a" />

    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:inputType="textPassword"
        android:ems="10"
        android:id="@+id/editText2"
        android:layout_below="@+id/editText"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_alignRight="@+id/editText"
        android:layout_alignEnd="@+id/editText"
        android:textColorHint="#ffff299f"
        android:hint="Password" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Attempts Left:"
        android:id="@+id/textView2"
        android:layout_below="@+id/editText2"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:textSize="25dp" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="New Text"
        android:id="@+id/textView3"
        android:layout_alignTop="@+id/textView2"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true"
        android:layout_alignBottom="@+id/textView2"
        android:layout_toEndOf="@+id/textview"
        android:textSize="25dp"
        android:layout_toRightOf="@+id/textview" />

    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_marginStart="-3dp"
```

```
        android:layout_marginLeft="-3dp"
        android:layout_marginBottom="15dp"
        android:layout_toEndOf="@+id/textview"
        android:layout_toRightOf="@+id/textview"
        android:text="Cancel" />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="4dp"
        android:layout_marginRight="4dp"
        android:layout_marginBottom="15dp"
        android:layout_toStartOf="@+id/textview"
        android:layout_toLeftOf="@+id/textview"
        android:text="login" />

</RelativeLayout>
```

MainActivity.java

```
package com.example.maids1;

import android.app.Activity;
import android.graphics.Color;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import android.content.Intent;
import com.example.maids1.myapplication.R;


public class MainActivity extends Activity {

    Button b1, b2;
    EditText ed1, ed2;
    TextView tx1;
    int counter = 3;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        b1 = findViewById(R.id.button);
        ed1 = findViewById(R.id.editText);
        ed2 = findViewById(R.id.editText2);
    b2 = findViewById(R.id.button2);
    tx1 = findViewById(R.id.textView3);
    tx1.setVisibility(View.GONE);

    b1.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
        if (ed1.getText().toString().equals("admin") &&
            ed2.getText().toString().equals("admin")) {
          Toast.makeText(getApplicationContext(),
              "Login in...", Toast.LENGTH_SHORT).show();
          Intent intent = new Intent(v.getContext(), Main3Activity.class);
          startActivity(intent);

        } else {
          Toast.makeText(getApplicationContext(), "Wrong Credentials",
Toast.LENGTH_SHORT).show();

          tx1.setVisibility(View.VISIBLE);
          tx1.setBackgroundColor(Color.RED);
          counter--;
          tx1.setText(Integer.toString(counter));

          if (counter == 0) {
            b1.setEnabled(false);
          }
        }
      }
    });

    b2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });
    }
}
```

Data Visualization & Action Control Activity: XML and Code

Activity_main3.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools = "http://schemas.android.com/tools"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    tools:context="com.example.maids1.Main3Activity">



    <WebView
        android:layout_width="fill_parent"
        android:id="@+id/webView"
        android:layout_above="@id/buttonlayout"
        android:layout_height="fill_parent" >

        <WebView
            android:id="@+id/webView2"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:layout_x="5dp"
            android:layout_y="400dp" />
    </WebView>


        <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
            android:orientation="horizontal"
            android:layout_width="fill_parent"
```

```
android:layout_height="wrap_content"
android:layout_alignParentBottom="true"
android:id="@+id/buttonlayout" >

<Button
   android:id="@+id/button1"
   android:layout_height="wrap_content"
   android:layout_width="wrap_content"
   android:layout_weight="1.0"
   android:layout_alignParentBottom="true"
   android:text="Retrieve Intrusion Database"
   android:drawableLeft="@drawable/dbicon3"
   />

<Button
   android:id="@+id/button2"
   android:layout_height="wrap_content"
   android:layout_width="wrap_content"
   android:layout_weight="1.0"
   android:layout_alignParentBottom="true"
   android:layout_toRightOf="@+id/button1"
   android:text="Retrieve RasPI Stats"
   android:drawableLeft="@drawable/statsicon3"
   />

<Button
   android:id="@+id/button3"
   android:layout_height="wrap_content"
   android:layout_width="wrap_content"
   android:layout_weight="1.0"
   android:layout_alignParentBottom="true"
```

```
        android:layout_toRightOf="@+id/button2"
        android:text="Test LED Module"
        android:drawableLeft="@drawable/ledicon3"
        />

    <Button
        android:id="@+id/button4"
        android:layout_height="wrap_content"
        android:layout_width="wrap_content"
        android:layout_weight="1.0"
        android:layout_alignParentBottom="true"
        android:layout_toRightOf="@+id/button3"
        android:text="Activate MAIDS"
        android:drawableLeft="@drawable/activateicon3"
        />

    <Button android:id="@+id/button5"
        android:layout_height="wrap_content"
        android:layout_width="wrap_content"
        android:layout_weight="1.0"
        android:layout_alignParentBottom="true"
        android:layout_toRightOf="@+id/button4"
        android:text="Deactivate MAIDS"
        android:drawableLeft="@drawable/deactivateicon3"
        />

  </LinearLayout>

</RelativeLayout>
```

MainActivity3.java Code

```
package com.example.maids1;

import android.annotation.SuppressLint;
import android.app.Activity;
import android.graphics.Bitmap;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.StrictMode;
import android.view.View;
import android.webkit.WebSettings;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Toast;

import com.example.maids1.myapplication.R;
import com.jcraft.jsch.Channel;
import com.jcraft.jsch.ChannelExec;
import com.jcraft.jsch.ChannelSftp;
import com.jcraft.jsch.JSch;
import com.jcraft.jsch.Session;

import java.io.BufferedInputStream;
import java.io.BufferedOutputStream;
import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.FileOutputStream;
import java.io.OutputStream;
import java.util.Properties;


public class Main3Activity extends Activity {

    private Button mButton1;
    private Button mButton2;
    private Button mButton3;
    private Button mButton4;
    private Button mButton5;
    private Button mButton6;
    private WebView webView;
    private ImageView imageView;
    private Bitmap bimage = null;
    private String username = "pi";
    private String password = "srgawain2264";
    private String host = "192.168.0.19";
    private int port = 22;
    private static JSch jsch = new JSch();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main3);
        if (android.os.Build.VERSION.SDK_INT > 9) {
            StrictMode.ThreadPolicy policy = new
StrictMode.ThreadPolicy.Builder().permitAll().build();
            StrictMode.setThreadPolicy(policy);
        }

        mButton1 = findViewById(R.id.button1);
    mButton1.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
        Toast.makeText(getApplicationContext(),
            "Retrieving DB Information...", Toast.LENGTH_SHORT).show();
        WebView htmlWebView = findViewById(R.id.webView);
        htmlWebView.setWebViewClient(new CustomWebViewClient());
        WebSettings webSetting = htmlWebView.getSettings();
        webSetting.setJavaScriptEnabled(true);
        webSetting.setDisplayZoomControls(true);
        htmlWebView.loadUrl("https://singular-gar-
5555.dataplicity.io/maidsintrusion.php");
      }
    });


    mButton2 = findViewById(R.id.button2);
    mButton2.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
        Toast.makeText(getApplicationContext(),
            "Retrieving RasPI Stats...", Toast.LENGTH_SHORT).show();
        WebView htmlWebView = findViewById(R.id.webView2);
        htmlWebView.setWebViewClient(new CustomWebViewClient());
        WebSettings webSetting = htmlWebView.getSettings();
        webSetting.setJavaScriptEnabled(true);
        webSetting.setDisplayZoomControls(true);
        htmlWebView.loadUrl("http://192.168.0.19:8080/");
      }
    });
```

```
        mButton3 = findViewById(R.id.button3);
        mButton3.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Toast.makeText(getApplicationContext(),
                        "Testing LEDs Remotely...", Toast.LENGTH_LONG).show();
                // Do something
                @SuppressLint("StaticFieldLeak") final AsyncTask<Integer, Void, Void>
execute = new AsyncTask<Integer, Void, Void>() {
                    @Override
                    protected Void doInBackground(Integer... params) {
                        try {
                            testLedsCommand(username,password,host,port);
                        } catch (Exception e) {
                            e.printStackTrace();
                        }
                        return null;
                    }
                }.execute(1);

                Toast.makeText(getApplicationContext(),
                        "LEDs Test Completed...", Toast.LENGTH_LONG).show();
            }
        });

        mButton4 = findViewById(R.id.button4);
        mButton4.setOnClickListener(new View.OnClickListener() {
            @SuppressLint("StaticFieldLeak")
            @Override
            public void onClick(View view) {
                Toast.makeText(getApplicationContext(),
                        "Activating MAIDS Remotely...", Toast.LENGTH_SHORT).show();
                // Do something
                final AsyncTask<Integer, Void, Void> execute = new AsyncTask<Integer,
Void, Void>() {
                    @Override
                    protected Void doInBackground(Integer... params) {
                        try {
                            maidsOnCommand(username,password,host,port);
                        } catch (Exception e) {
                            e.printStackTrace();
                        }
                        return null;
                    }
                }.execute(1);
            }
        });

        mButton5 = findViewById(R.id.button5);
        mButton5.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Toast.makeText(getApplicationContext(),
                        "Deactivating MAIDS Remotely...", Toast.LENGTH_SHORT).show();
                // Do something
                @SuppressLint("StaticFieldLeak") final AsyncTask<Integer, Void, Void>
execute = new AsyncTask<Integer, Void, Void>() {
                    @Override
                    protected Void doInBackground(Integer... params) {
                        try {
                            maidsOffCommand(username,password,host,port);
                        } catch (Exception e) {
                            e.printStackTrace();
                        }
                        return null;
                    }
                }.execute(1);
            }
        });
    }

    class CustomWebViewClient extends WebViewClient {
        @Override
        public boolean shouldOverrideUrlLoading(WebView view, String url) {
            view.loadUrl(url);
            return true;
        }
    }

    public String testLedsCommand(String username, String password, String host, int
port)
            throws Exception {
        //JSch jsch = new JSch();
        Session session = jsch.getSession(username, host, port);
        session.setPassword(password);
        // Avoid asking for key confirmation
        Properties prop = new Properties();
        prop.put("StrictHostKeyChecking", "no");
        session.setConfig(prop);
        session.connect();
        // SSH Channel
        ChannelExec channelssh = (ChannelExec) session.openChannel("exec");
        ByteArrayOutputStream baos = new ByteArrayOutputStream();
        channelssh.setOutputStream(baos);
        // Execute command
        channelssh.setCommand("python3 ~/testleds.py");
        channelssh.connect();
        channelssh.disconnect();
        return baos.toString();

    }

    public String maidsOnCommand(String username, String password, String
hostname, int port)
            throws Exception {
        //JSch jsch = new JSch();
        Session session = jsch.getSession(username, hostname, port);
        session.setPassword(password);
        // Avoid asking for key confirmation
        Properties prop = new Properties();
        prop.put("StrictHostKeyChecking", "no");
        session.setConfig(prop);
        session.connect();
        // SSH Channel
        ChannelExec channelssh = (ChannelExec) session.openChannel("exec");
        ByteArrayOutputStream baos = new ByteArrayOutputStream();
        channelssh.setOutputStream(baos);
        // Execute command
        channelssh.setCommand("python3
~/maids_final_python_code_22102019_bak1.py");
        channelssh.connect();
        channelssh.disconnect();
        return baos.toString();
    }

    public void maidsOffCommand(String username, String password, String hostname,
int port)
            throws Exception {
        //JSch jsch = new JSch();
        Session session = jsch.getSession(username, hostname, port);
        session.setPassword(password);
        // Avoid asking for key confirmation
        Properties prop = new Properties();
        prop.put("StrictHostKeyChecking", "no");
        session.setConfig(prop);
        session.connect();
        // SSH Channel
        ChannelExec channelssh = (ChannelExec) session.openChannel("exec");
        ByteArrayOutputStream baos = new ByteArrayOutputStream();
        channelssh.setOutputStream(baos);
        // Execute command
        channelssh.setCommand("python3 ~/reboot.py");
        channelssh.connect();
        channelssh.disconnect();
        //return baos.toString();
    }



    }
```