Smart Surveillance Hub

Joshua Sherrill, Korey Lombardi, Kenneth Ancrum, Matthew Weinert

Dept. of Electrical Engineering and Computer Science, University of Central Florida, Orlando, Florida, 32816-2450

Abstract — **The goal of this project was to design a smart surveillance hub that notifies the user when a human is detected. The product provides notifications when a user is detected and offers a level of security to the home. Any purchased IP camera on the market is easily connected to the hub via a RTSP stream and able to be streamed. The microcontroller is responsible for handling all the notifications such as light control and audio control. The Raspberry Pi is responsible for running the human detection model. The Raspberry Pi detects a human and sends a signal to the microcontroller which then causes the lights and audio to change from their idle state into a state of alert. The camera is able to be viewed in real time via the led screen. This is the start of a product that could be easily used for other smart home devices.**

I. INTRODUCTION

Privacy and security are a standard in today's society, and the ability to keep yourself and your family safe should be accessible and affordable. According to the FBI's Uniform Crime Reporting (UCR) Program, the number of burglaries has decreased 48.5 percent from 2010 to 2019. Many systems on the market require large upfront cost or lengthy contracts. The Smart Surveillance Hub is a cheaper alternative to traditional security systems. With the Smart Surveillance Hub users are able to self monitor their home at a lower cost with a more flexible system. The use of the RTSP stream of the camera allows the user more flexibility when purchasing cameras. Our project's main task is to provide safety for individuals or families when they are located inside their homes and provide them with enough time to respond accordingly. Whether the user wants to have a bird's eye view of their front door simply or if they're going to use it to identify potential intruders, it's a simple way to be alerted without being pestered. Although our product will secure the homes of users while they are located inside their homes, it won't provide them with footage when they are not located inside their homes.

The project has two major components, the software and the hardware. The software is responsible for an AI detection model which will continuously scan the camera footage and look for humans, once a human is detected the software sends a signal to the hardware and then two actions are carried out through the hardware. The software on the main application will allow the user to utilize a bunch of features from storing previous recordings, fast forwarding past recording, rewinding, and pausing. All of these features can be used from the LCD screen.

II. ENGINEERING SPECIFICATIONS

Our highlighted specifications include a smart hub desktop application the has RTSP connectivity speed of under 5 seconds. We also wanted to meet standards that you sould find on the market so another specification is to have a speaker system that can produce a sound sensitivsity rating of 80dB or more. We also wanted our system to utilize A.I. that can detect humans in up to 5 seconds. For housing and assembly the 3D printed shell will need to be less than five pounds and able to to comrotably fit all needed components.

Engineering specification	Vlaue rating
RTSP connection time	< 5s
Human detection time	< 5s
Speaker sensitivity rating	> 80dB
Overall system weight	< 5lbs

III. GOALS

Our overall objective is to give the user all the power and not the system. Security is essential because it gives people the ability to prevent dangerous situations and be safe. Many current systems are not priced friendly for everyone, which is an issue. Safety and security should not be only for those who can afford them. Our design aims to give the user everything they need to use whatever camera they want and be safe while observing the camera feeds or being away from them. Our specifications go into a bit more detail on the desired deliverables from the system while ultimately being affordable and customizable to everyone. By building our design with efficient and low-cost components, it is possible to provide security for users of all types of incomes. It will also benefit business owners who want to secure their business more realistically than by subscribing to a well-known security company that will charge them more than necessary. We feel that everyone from all walks of life and ranges of income should feel safe and confident in their home without paying large amounts of money.

The three main objectives we wanted to obtain in this project. The first goal is to be able to connect at least 3 different IP cameras to the system, this will show that our project is versatile and can handle a range of cameras on the market.

The second goal is to deliver a functioning human detection system, which is able to detect a human of up to 20 feet. 20 feet is practical distance when it comes to home security because a majority of front yards/backyards do not need detection of humans further than that.

The last goal we hope to achieve with this project is to deliver an accurate detection within 5 seconds of a user entering the frame of the camera. With the current hardware we are using we calculated that a 5 second delay is an acceptable delay for an alert system.

IV. HARDWARE OVERVIEW

We wanted to make a system as practical and sensical as we could. To do this we researched and thought about some features and technologies that people would expect or want in their own security surveillance system after careful consideration. These are the hardware choices that we have landed on. These components and technologies were chosen to offer a powerful, lower power, robust system that could be used for practical use.

A. LED Lighting

The versatility and options with these strips are endless. LED strips consist of a series of red, blue, and green light emitting diodes that in series with each other can be controlled to create color. For traditional LED 4 port LED strips a current flows through the diode and light is produced, when a different amount of light is produced in each of the 3 diodes different colors are generated. We decided to choose a more efficient 3 port LED strip. 3 port LED strips have a port for ground,5v, and data in . Using this configuration the LEDS can be easily programmable. It would also make voltage distribution easier since we can control the supply of the 5v rail better amongst all pixels on the strip. LEDs have many advantages over different light sources due to the low power consumption and the life of the diodes. LEDs require a direct current (DC) because a pulsating AC wave does not allow the diode to emit enough light consistently.

B. Audio amplifier

An audio amplifier is an electronic amplifier that amplifies low-power electronic audio signals to a level that is high enough for driving loud speakers or headphones. Audio power amplifiers are found in all manners of sound systems and are typically the final stage of the audio playback chain before the signal is sent to the loudspeakers. We will need an audio amplifier if we use a DAC for emitting a sound notification. The audio amplifier must produce a sound loud enough that the user of our system can hear it from a distance or if any background noise exists.

For our project the Amplifier will be powered by the 5 volt line of the USB connector and will take left and right stereo signals from the PCM122 digital to audio converter. Our amplifier needed to be able to be low power and stereo. It also needed to have a decibel rating of about 85-90dB when driving 4ohn 3watt speakers. For these spsecifications we selected the MAX98306ETD+T wich features pop suppression and Output Power of 3.7W at 3Ω. The amplifier will drive 2 40mm speakers to supply efficient noise levels to fill a room on quede alert.

C. Digital to audio converter

A digital to analog converter takes a digital signal and converts it into an analog audio signal. Any system that involves the playing of an audible sound sent from a computer requires a DAC. Without the use of a DAC the sound is merely just a collection of bits, a collection of 1's and 0's. The DAc we needed had to be able to take I2S communication, for the left right clock, bit clock and data in connections for simple wiring. The use of I2S also allow us to excuse any unnecessary EMI that may cause issues with our data lines

The PCM5122 by texas instruments was our choice. It provides for CM Interface and Fixed Audio Processing With a two channel configuration,and a 32 bit resolution bus. This will meet our criteria for expanding our audio capabilities from our microcontroller. The Integrated circuit also uses the I2S Pulse Code Modulation to generate an appropriate sound frequency Using the equation below we can select values for our data lines to generate the sound frequency we want, for example since most mp3 files operate at 44.1KHz, we can choose values for the data lines to generate said signal.

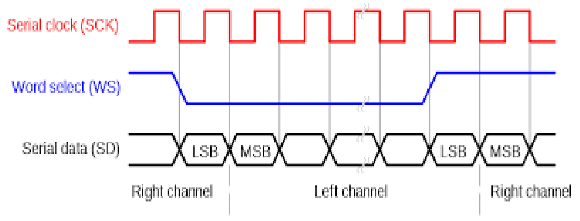


Fig. 2. I2S Freuqeucny

BCLK = WS * Word count * Word length

The DAC also has the ability to be I2C controlled and operated giving us the option of features such as volume control and the addition of a master clock for host mode operation. We decided to keep the DAC in a slave mode operation as it will be commincating with our RP2040 Microcontroller and connection is less prone to EMI risks.

D. USER DISPLAY

Since the Raspberry pi 4 module has the ability to accept a Display serial interface we considered this a display option for lowering the cost of our overall build. The DSI model was a specification introduced by the Mobile Industry Processor Interface (MIPI) Alliance,This is a global tech alliance that specializes in developing specifications for the mobile ecosystem. Using DSI is our best option for low power consumption, since the MIPI interface uses low voltage signaling which has the benefit of low power operation. But even at such a low power it can transmit at a wide range data transmission allowing for less interference of other peripheral devices while allowing for less pins and offering better performance than other interfaces.

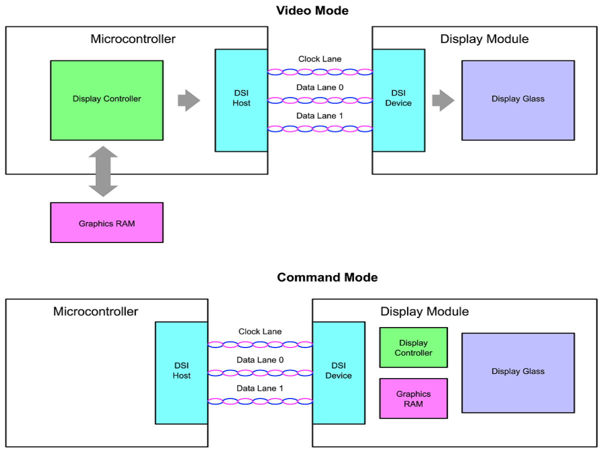


Fig. 3. DSI Diagram

The interface operates with a differential signaling point-to-point serial bus on the physical layer with the latest with the latest lanes for D-PHY operating at speeds of 4.5 Gbps/lane. The Bus consists of one high speed clock lane and one or more data lanes where each lane is carried on a two wire configuration. All lanes will bridge between the DSI host and device where the lanes will transmit in parallel, meaning if four lanes are being used then all 4 bits will be transmitted simultaneously. There are two modes that can be used: the video mode and the command mode.

The video mode will be the standard operation for most systems. In Video mode RGB pixel data and horizontal and vertical sync signals provided by the display controller are encoded into the serial stream by the DSI Host (in our case this would be our raspberry pi 4 microcontroller) and decoded by the Device. Alternatively, the display controller and graphics RAM can be integrated into the display, this can take the load off the DSI host as its only responsibility would be to transmit the pixel data. These functions make the MIPI DSI interface a lot more complex than the classic parallel RGB plus clock and sync signals, but offers High performance, Low power, and Low EMI.

V. HARDWARE DESIGN

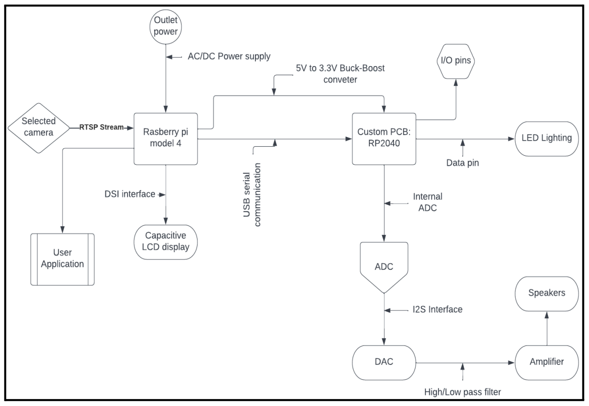


Fig. 4. Hardware diagram

The Surveillance Hub is split into two systems: the system of our own PCB design and the system of peripherals that connect to the central unit that will consist of our DSI LCD screen and the External cameras. The other part of our system will be controlling and communicating with the PCM5122, Sound Amplifier, and lighting. As well as communicating with the central half.

There will be two microcontrollers used to operate the system. These two separate microcontrollers will communicate through serial USB, while the central unit will have network capabilities and transfer data to our application as well as have higher computation and processing power. For this system, we will use one Raspberry Pi 4 and an RP2040.

Since the central unit is responsible for alot of our systems higher computation and logic we the raspberry pi 4 is a good choice for us. The raspberry pi features a Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz. This is not the most powerful CPU in the world but it is good for use in our project. It also has 8GB LPDDR4-3200 SDRAM and comes with many usb ports including a LAN port which is crucial to our overall project.

For the other part of our system, we will be incorporating the RP2040 Microcontroller in our PCB design.The RP2040 is a 32-bit dual ARM Cortex-M0+ microcontroller integrated circuit by Raspberry Pi. It features 133 MHz dual ARM Cortex-M0+ cores for fast processing, which meets our specifications, the ability to have a max Flash Memory capacity of 16MB for expanded use, and 30 GPIO pins for added peripherals. The RP2040 is the brains of our lighting and sound function and communicates with Raspberry pi 4 both through serial and through UART.

VI. POWER SYSTEM

Due to a comprehensive relatively low power system we were able to generate all the power on the PCB via the Raspberry Pi. The issue that needed to be solved was regulating the power from the Raspberry Pi to the microcontroller. Multiple power converting circuits were needed in order to create a steady current that did not fry our system, the most important being a buck converter. The first buck converter we used was from the Raspberry Pi to the microcontroller which allowed us to step down the 5 voltage output to the 3.3 volts needed for the microcontroller. The buck converter allowed us to regulate the voltage from the microcontroller to the audio system. By utilizing a low noise regulator we were able to ensure the audio quality was unaffected by the microcontroller. Under the system specs the Raspberry Pi used most of the power in our system and required 2.875 Amps to operate. The PCB required much less power at roughly 1.8 Amps. In order to power the Raspberry Pi we needed a direct contact to an outlet. Below is a figure of the buck converter we used for the low noise lines.

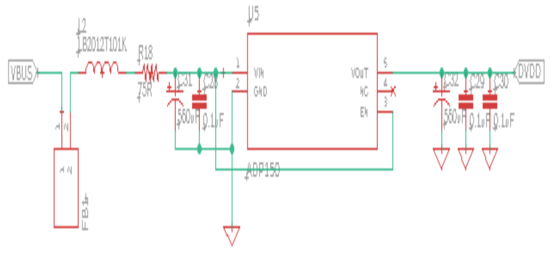


Fig. 4. ADP150 Voltage Regulator

VII. BOARD DESIGN

This project falls under a dedicated electronic service and falls under general electronic products. In order to start the design process we needed to study the PCB standards. The standard needed for this process is IPC-2221, which offers the basic standards regarding any PCB. We used the Autodesk software EAGLE in order to design the schematic and board layout. This software was free to us and was most convenient since we have had experience using the software. Eagle allowed us to generate multiple schematics and find the available parts required for completing the PCB design. Below is the board layout that was sent to be printed.

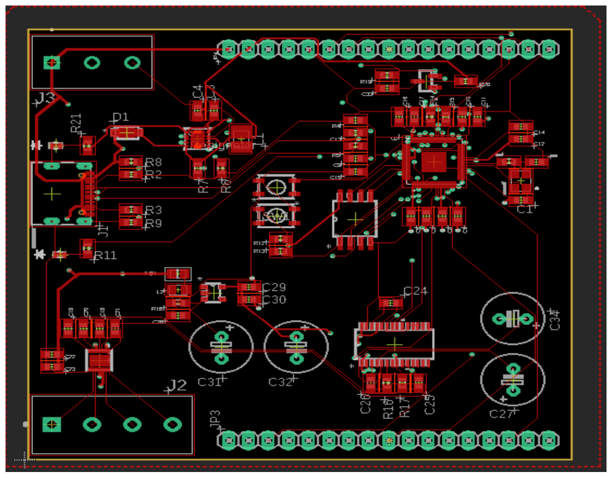


Fig. 5. Board design layout

VI. CASE DESIGN

One of the most important aspects to a surveillance hub is the hub itself, for the design of the hub we used fusion 360. The screen will be screwed into the front of the case. The speakers will be screwed in at the bottom of the case, the LEDs will wrap around the case inside a light diffuser for maximum visual effect. The back will be removable for easy access to the interior components but can be sealed easily. Finally the PCB will be attached to the right side of the case via shelf clips. Below is a picture of the 3D modeled case.

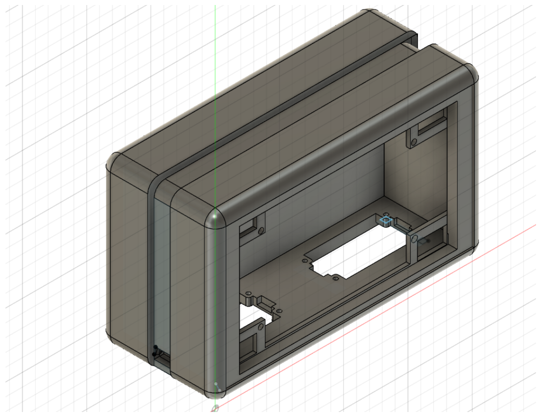


Fig. 6. Shell Case Design

VI. SOFTWARE DESIGN

A. Software Overview

For the following sections in software design, we will go over the system on a higher level. We will go over some of the more common protocols, such as the most commonly used one in the software, the Real Time Streaming Protocol, otherwise commonly known as RTSP. Ultimately the goal of this section is to link the connection between the camera, our raspberry pi which is acting as the main component and the pcb notification device.

B. Network

Overall the deliverable is to create a system that handles a video camera operating on a network using the internet as the median. In order to make that happen, we can look at the firmware and software from one endpoint to another. The camera first needs to be connected on the internet, which means the camera must require WiFi or ethernet connection capabilities. Typically an off the shelf camera, which is the type of camera we aim to interface with, requires its own firmware to be recognizable on the network's ports, and that not every IP camera supports this software functionality but most do. This functionality gives the network recognizable ports to stream the cameras feed off of.

This protocol exists on the application-level which allows us to communicate across the network with real-time data which is going to be the live security feed of what the camera is picking up. This data is going to be a large data size, and because of that prior to being sent over the network, it will be encoded. Encoding the data, more commonly known as video compression, will ensure that the size of the data sent over the network yields a lower throughput and will result in less delay between the time of recording and the real time.

The next step in the chain at the other end of the network is the SmartHub application. With RTSP, in order to interface with the camera itself the only requirement is that the SmartHub needs to be on the same network as the camera. This is because RTSP gives access to the camera's feed on the network using a TCP port.

C. RTSP

So now we established the communication connection between the camera and the SmartHub, we can go a little bit into RTSP. In order to access the RTSP stream from the camera, RTSP requires the camera to have a username and password so not anyone on the network can easily access the feed which is a subtle layer of security. Also since you need to be on the network in order to access the ports to begin with, that is another pseudo layer within our network design in the name of security.

The RTSP string itself is an all-in-one string that contains multiple parsable pieces of information.

A generic RTSP stream URL, like the one used to connect to one of the camera's on the network is like such:

```
rtsp://admin:123456$@192.168.1.243:554//h264Preview_
01_main
```

Fig. 7. RTSP stream required for accessing camera

Looking at the URL, there are a few things that we can go over. This is a parsable string in our case with four main pieces of data that we need to ensure a connection. The first thing is 'admin', this is the username associated with the IP camera itself. The Reolink camera we have is defaulted to admin, but we have the capability to change this username with Reolink's own application in their camera's network configuration section. Each camera has their own configurable constants. There is a colon separator between the username and the password being the 123456, which is also a configurable password. For the sake of the project it was easy to make it something short and simple rather than something secure, but again the password is also very easy to change.

The next part of this URL is the IP address along with the port where the IP camera is ported to. The IP address is a private address meaning that it is not directly exposed to the internet, but rather to the network itself. In order to access this address you must also be on the network that the address exists on, and this is also true with the port 554. Lastly there is the server url, which is different per IP camera. Each time RTSP is used, we have its own relative server. This server is responsible for handling the TCP connection and configuration between the client, being our SmartHub and the camera. This software is firmware that is pre-installed into the cameras and is out of scope for our use.

There are additional optional parameters that an RTSP stream could use, and that is parsed on the server to do additional optional functionality, but for our system we did not need or have those options. Some of these options can be optional compressing or changing the compression type. Like mentioned before, we are compressing the data using the H264 codec to ensure speed and it is decompressing the data on the pi using the same codec.

VIII. SOFTWARE STACK

A. Software Utilization

Up to this point we covered the network communication and the RTSP communication over the network with some introduction on how everything is chained together. At this point in the overview we established the connection between the endpoints and we can go into more specifics on the software application itself.

After attempting a few other software stacks that were underdeveloped, our goal was to create a stack that had enough frameworks and libraries to offer support for what we needed. Originally the NodeJS stack we were going to use was under-developed and didn't offer all the functionality we needed, so we stitched together a Python stack full of all the functionality we needed.

B. Software Stack

The first framework being used is PyQt5 which is a cross-platform GUI, similar to a Java Swing application, which offers component and widget customization on a very basic level. Along with PyQt5 we use OpenCV for image processing. OpenCV has open support for RTSP streams, meaning when generating the stream and connecting to the camera in the software we simply need the RTSP URL to configure the connection in which we render the video feed to on our widget. Each connection to the camera is threaded in order to maximize system utilization. Along with being able to stream RTSP videos using OpenCV, we also use its provided human detection models to trigger a conditional to send the PCB notification device a signal to carry out the PCBs features.

So ultimately, the stack consists of two main frameworks which are PyQt5 and OpenCV along with a few additional libraries, like python's threading library or their virtual keyboard for the touch screen to have a keyboard that is displaying the application.

C. Software Features

The goal for the software was to establish an error prone, simple and solid way to provide a user with a way to connect to the camera and display the live feed. On top of this we have additional features that we created off of the bare-bones of the framework. The largest other feature is recording previous stream footage and saving it locally for playback purposes. Prior to a user's connection, they have the option to toggle this feature off if they do not want to record the stream. When establishing a connection to the stream, that is when the software creates a .mp4 file where it writes and appends each frame to. After establishing a video connection, they can watch the stream in real time and if the camera picks up a security threat, which is simply recognizing a human, it will alert the user on the hub which will produce a sound alert along with changing the mood lights to red. When the system stops recognizing a human, the lights will turn back to blue. The sound alert only plays initially and is not continuous and will play again after the state changes to safe, and then detects another threat.

With the live stream being recorded, the user can also opt to watch the recording instead of the live stream. The GUI contains a list of all previous recordings, with the names defaulting to a format we decided, which is *MonthDayYear - HourMinuteSecond* of when the recording initially started. Along with just watching the stream, the user has buttons to watch the recording, such as fast-forward, rewind, play and pause. Below is a higher level logic diagram that describes the two main features and what the logical differences between the recording and live stream features are.

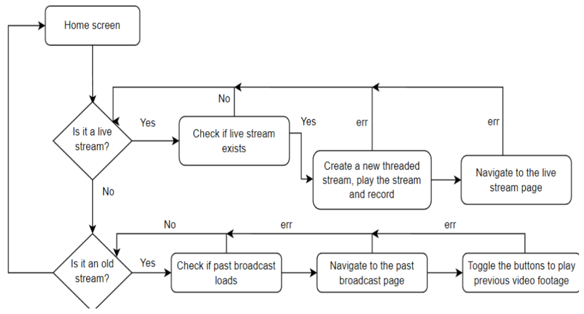
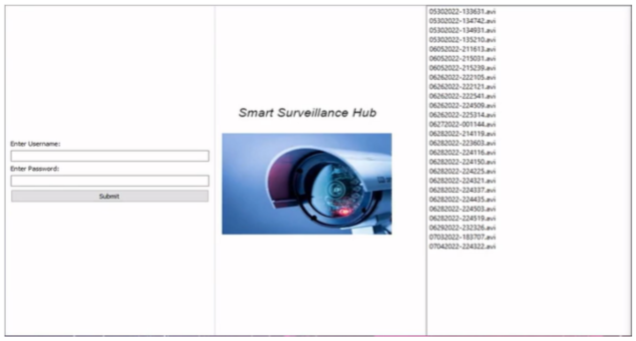

Home screen
Is it a live stream?
Yes
No
Check if live stream exists
Yes
err
err
Create a new threaded stream, play the stream and record
Navigate to the live stream page
No
err
err
Is it an old stream?
Yes
Check if past broadcast loads
Navigate to the past broadcast page
Toggle the buttons to play previous video footage

Fig. 8. Application's workflow

IX. USER INTERFACE



```
05302022-133631.avi
05302022-134742.avi
05302022-134931.avi
05302022-135210.avi
06052022-211613.avi
06052022-215031.avi
06052022-215239.avi
06262022-222105.avi
06262022-222121.avi
06262022-222541.avi
06262022-224509.avi
06262022-225314.avi
06272022-001144.avi
06282022-214119.avi
06282022-223603.avi
06282022-224116.avi
06282022-224150.avi
06282022-224225.avi
06282022-224321.avi
06282022-224337.avi
06282022-224435.avi
06282022-224503.avi
06282022-224519.avi
06292022-232326.avi
07032022-183707.avi
07042022-224322.avi
```

Fig. 9. Applications main window

The user interface provided by our application was designed to provide the user with a simple way of navigating through the application's features and maximizing functionality. The main window is separated into three distinct components. When building the user interface, we wanted the user to feel comfortable switching between windows without getting lost in the application.

Initially, the user is brought to the main window, where they can enter their RTSP stream credentials to access the camera's footage. The values that will be input from the user consist of username and password. Once the application receives the credentials, it concatenates the RTSP string using the credentials given by the user. Once the application receives the correct credentials, the user is moved to another window where the camera's footage is displayed.

The second part of the main window is a photo of our company's logo in the middle of the screen. This part of the interface does not provide any functionality but is there to create a professional look and fill in any blank space.

The third component of our main window allows the user to access the older camera footage from previous sessions. The recordings are sorted by date and stored locally on the Raspberry Pi. When the user clicks on one of the recorded video files, the user is brought to a window displaying the footage giving the user the ability to pause, fast forward, and rewind.

Once the user navigates to a particular app feature, they will be presented with a back button to ensure the user never feels they are stuck at any window. We also ensured that all controls and clickable items were large enough for the user to click without interference from other buttons or features.

X. TESTING

We believe the two most important tests include verifying the human detection accuracy rate is above 95% and keeping the time it takes for the RTSP to establish a connection to the app's live video feed. After conducting the same test ten times, we verified the human detection accuracy rate to be 100%. We understand it's nearly impossible to achieve 100% accuracy, but since we only ran the test ten times, we achieved sufficient results. Below are the results for the time it took us to establish the RTSP stream connection and display the footage.

Mean	Min	Max
3.06s	2.9s	3.5s

Fig. 10. Testing results for RTSP stream connection time

The testing results for the the times it takes the detection software is also below ans is based on 10 testing values.

Mean	Min	Max
2.62s	2.0	3.1s

XI. CONCLUSION

Our senior design project ultimately was a challenge, assembling a group that didn't know each other prior to this course. Luckily we all share the skills required to carry out a project with high complexity, and all teammates learned some new engineering techniques and practices throughout the build process.

A significant challenge for us as a group was agreeing on the project's scope. Throughout the build process, we faced various issues regarding the complexity of the design. Another challenge was ensuring that the components and algorithms used for the design maximize the performance by choosing the parts or technology with the most beneficial tradeoffs.

We can all say we have learned a lot regarding time management and communication. It's been helpful that our team has been conscious of our time management to get things done on time. We also learned how to communicate ideas efficiently by agreeing on a solution by comparing its tradeoffs. With the skills we acquired by building this project, we feel we are now prepared to enter the professional electrical and computer engineering industry.

XII. THE TEAM

Kenneth Ancrum is currently a senior at the University of Central Florida and will be graduating in August of 2022 with a major in electrical engineering. And hopes to go into the industry to drive innovation forward. Kenneth is most knowledgeable in hardware and controls topic and wishes to work in with hardware and design after graduation. Kenneth is responsible for hardware research and overall PCB design and assembly.

Korey Lombardi is currently a senior at the University of Central Florida and will graduate in August of 2022 with his degree in computer engineering. He is passionate about building products from scratch, and Korey is most comfortable in the web development ecosystem. After graduation, he plans to start his own freelance software company specializing in restaurant software.

Joshua Sherrill is currently a senior at the University of Central Florida and will be graduating in August of 2022 with a major in electrical engineering. Joshua has a strong background in sales and experience with technical products. Upon graduating Joshua hopes to enter the technical sales field where he is able to leverage his outgoing personality and his technical ability to push innovative products to market. Joshua has a strong understanding of the market and hopes to become a sales engineer after graduation.

Matthew Weinert is currently a senior at the University of Central Florida and will be graduating in August of 2022 with his degree in computer engineering. While in school, Matt has accumulated nearly 2 full years of professional Software Engineering experience, across 3 different federal contractors and 4 different contracts. Currently he has a full time position as a full-stack Software Engineer at 4C Strategies and his goal is to be an adjunct professor or run a software startup on the side in his free time post-graduation.

XIII. References

[1] Federal Bureau of Investigation. (2019). *2019 Crime in the United States: Burglary.* https://ucr.fbi.gov/crime-in-the-u.s/2019/crime-in-the-u.s.-2019/topic-pages/burglary

[2] ADT. (2022). *Is a Wireless Security Camera Worth the Cost?*. https://www.adt.com/resources/wireless-security-cameras-cost

[3] Raspberry Pi 4 Specification, Schematic, Diagrams https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/

[4] Teske, Lucas. June, 2019. *Reverse Engineering cheap chinese "VRCAM" protocol.* Medium. https://lucasteske.medium.com/reverse-engineering-cheap-chinese-vrcam-protocol-515c37a9c954

[5] Vivint. 2022. https://www.vivint.com/ppc/brand?cq_src=google_ads&cq_cmp=146236557&cq_con=7629216237&cq_term=vivint&cq_med=&cq_plac=&cq_net=g&cq_plt=gp&utm_medium=cpc&utm_term=vivint&utm_source=google&c_ps=s.google_k.vivint_m.e_n.g_po._d.c&exid=117156&geo=9053023&gclid=Cj0KCQiA64GRBhCZARIsAHOLriIg_qJ0Kg_h_ZH3ORTHyE4MRUnbkoV8cjiA0kLyc_uWDAdT0i54SyUaAtDzEALw_wcB

[6] Ring. 2022. https://ring.com/collections/all-products

[7] ADT. 2022. https://www.adt.com/security-cameras

[8] Jest. 2022. https://jestjs.io/

[9] JavaFX. 2022. https://openjfx.io/

[10] Eclipse Foundation. 2022 *The Eclipse Jetty Project.* https://www.eclipse.org/jetty/

[11] S. Abdelgawad, K. Yelamarthi. 2019. *IoT Based Security System: Pre-College Research Project.* http://people.se.cmich.edu/yelam1k/asee/proceedings/2019/1/3.pdf

[12] J. Wang, H. Ping, M. Fang. 2019. *Home Security System Using Raspberry Pi.* https://people.ece.cornell.edu/land/courses/eceprojectsland/STUDENTPROJ/2018to2019/jw2527_hp394_mf734/project_142_report.pdf -better-choice/

[13] PCMag. 2022. https://www.pcmag.com/encyclopedia/term/mp3

[14] Texas Instruments. 2022. PCM5122. https://www.ti.com/product/PCM5122

[15]Texas Instruments. 2022. PCM5102A. https://www.ti.com/product/PCM5102A

[16]analog devices.2022. ADP150. https://www.analog.com/en/products/adp150.html

[17]**[**Texas Instruments. 2022. TPS63001 https://www.ti.com/product/TPS63001?utm_source=google&utm_medium=cpc&utm_campaign=app-null-null-GPN_EN-cpc-pf-google-wwe&utm_content=TPS63001&ds_k=TPS63001&DCM=yes&gclid=Cj0KCQjw2_OWBhDqARIsAAUNTTFk1onzqfqkLe8BCpHtk_ARoEiKlkzcJ0MYNdl-cb6WB2y-QfV5-hEaAgJGEALw_wcB&gclsrc=aw.ds

[18]Newark. 2022. MAX98306ETD+T. https://www.newark.com/maxim-integrated-products/max98306etd-t/audio-amp-class-d-3-7w-tqfn-14/dp/73Y6584

[19]. *Raspberry Pi Datasheets*. https://datasheets.raspberrypi.com/pico/pico-datasheet.pdf.

[20]*ATmega 32 Datasheet*. https://ww1.microchip.com/downloads/en/DeviceDoc/doc2503.pdf.

[21]*ESP32S3 Series - Espressif.* https://www.espressif.com/sites/default/files/documentation/esp32-s3_datasheet_en.pdf.

[22]*Raspberry Pi RP2040 Datasheets*. https://datasheets.raspberrypi.com/rp2040/rp2040-datasheet.pdf.