ILLINOIS INSTITUTE OF TECHNOLOGY

Through a Speculum

- Smart Mirror -

Final Report

ECE 441: Smart and Connected Embedded Systems

Advisor: Prof. Jafar Saniie

PREPARED BY

Alan Palayil - B.S. Computer and Cybersecurity Engineering

Fabian Garcia - B.S. Computer Engineering

Gabriel Gutierrez - B.S. Electrical Engineering

Table of contents

Our Team

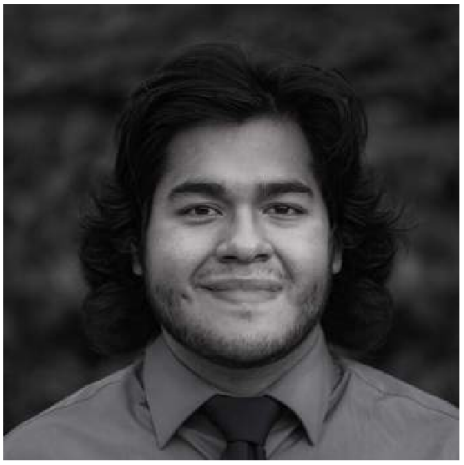
Alan Palayil

B.S. Computer and Cybersecurity Engineering

A Computer and Cybersecurity major who is focused on the software integration and management for the Smart Mirror project. In the past, he has worked on a few IOT projects where he integrated IOT hubs and built smart home proposals. The smart home included cloud connectivity and security monitoring. Alan is currently enrolled in the coterminal program for M.S. in Cybersecurity Engineering.

Gabriel Gutierrez

B.S Electrical Engineering



An Electrical Engineering major who focused on the hardware design and enclosure creation for this Smart Mirror design. In the past, he has done research on a photovoltaic grid-tied system where he researched and simulated an optimal 120V/60Hz DC to AC solar power system. Gabriel has been admitted to the accelerated master's program where he will specialize in communication theory and signal processing.

Fabian Garcia

B.S. Computer Engineering

A Computer Engineering major who focused on the construction and implementation and troubleshooting of computer vision and camera-based functions of the mirror. In the past, he has worked on the creation of a mechanical arm for handicapped individuals that will allow them to lift more than they are capable of due to injury. The mechanical arm included design work, implementation of code, and mechanical components.

Abstract

The Speculum Smart Mirror is not only a tool that will be used on a daily basis for checking one's appearance. Rather than providing that limited functionality, our smart mirror offers entertainment and valuable information through various widgets and display configurations, ultimately creating a helpful device that can be easily integrated into a daily routine and lifestyle. The entertainment aspect of the device is achieved through a speaker system that functions both through a wired connection and a Bluetooth connection. The speakers work in conjunction with the display and LEDs to provide an immersive music experience that can be controlled directly from your phone or the on-screen widget. As for the user interface, the UI will be handled through user touch inputs or specific gestures that will be seen through a camera module on the mirror and processed through our Raspberry Pi and specific program to achieve the desired actions without physically touching the mirror's surface.

Introduction

This smart mirror was designed to create a fun and interactive mirror. It has the ability to display the calendar, play music, show the current news, and act as an away from the home security system. It was designed using easily obtainable parts which help provide the user with a high-quality experience. This system's main component is the Raspberry Pi 4, which serves as the hub for all the features such as the microphone, speaker, camera, and the Android 12 operating system.

This report will cover the components of the smart mirror, such as the hardware and the importance of the Raspberry PI, the software that was used, and the decision that went into deciding the user interface. With the combination of these features, the user can experience a unique experience while having the option to use the device as an everyday mirror.

System Specifications

Standards

USB

The Universal Serial Bus is an industry-standard that allows for communication, power supply, and connections between different devices There are currently 4 generations of USB, however, the team focused on 2.0 and 3.0 since this is what's supported by the Raspberry Pi 4,

High Definition Multimedia Interface

The HDMI allows for uncompressed video data and digital data to be transferred

IEEE 802.15.1 - Bluetooth

Standard IEEE 802.15.1 refers to the wireless communication technology Bluetooth. This is meant for low-range communication between various devices. It is broken into

Class 1, 2, and 3 distinguished by their range of 100, 10, and 1 meter. It operates in the same 2.4 frequency as WLAN/WIFI

IEEE 802.11 - WLAN/Wi-Fi

Wireless LAN which is also known as wifi is a data communication standard that operates within the 2.4 and 5 GHz range. It is widely used in the consumer market.

System Constraints

Cost

- An extravagant smart mirror can be produced using high-end microphones, speakers, and even a monitor; however, to maintain a cost-effective solution for this prototype, the team decided to choose and reuse lower-end components.

Computational power

- In order to maintain a compact mirror, a Raspberry Pi was selected to serve as the brain of this project. The following are some constraints:
    - Requires an SD card for storage
    - No graphics processor
    - Slow processing speed compared to a traditional computer.

  Although the Raspberry Pi 4 presents some challenges, it is still a powerful tool when it comes to designing small-scale projects.

Enclosure space

- Since this smart mirror has the ability to compute with a traditional mirror, it must also be compact and moveable. It must house all the components (see **Hardware Design** for a full description) in a 14 by 22-inch enclosure with a 2-inch depth. This enclosure must also include ventilation to prevent components from overheating in a compact space.

User Interaction

- Although the Raspberry Pi can use keyboards and mice, the user will be limited to the use of a touch IR frame and any tools supported by Google's Voice assistant. Further modifications may be done to incorporate the mouse and keyboard, but such methods were pursued in this project.

Hardware Design

Electronics

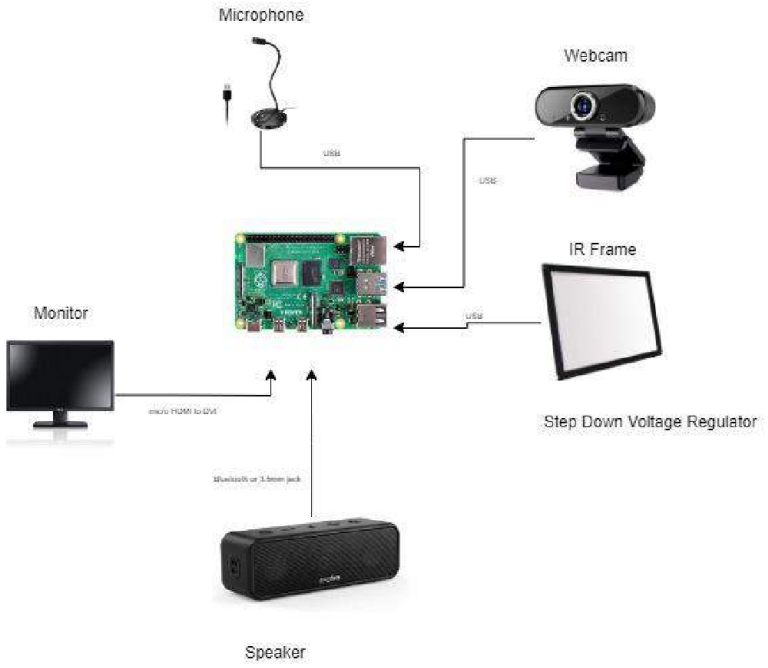


Figure 1: Schematic of the smart mirror electronics

Raspberry Pi

The Raspberry Pi 4 4GB extreme kit was selected for this project. This kit provides a 128GB SD card, three heat sinks, a fan, an enclosure, an HDMI cable, and a 3.0V power supply. A feature that differentiates it from its Pi 3 counterpart is the USB 3.0 port which allows additional devices to be added. Cheaper methods were sought out but the current Raspberry Pi boards are in high demand and low in stock, so the kit mentioned above was chosen.

Speaker

The speaker chosen for this project was the Soundcore 3 Anker speaker. This speaker supports audio through a Bluetooth connection and a 3.5mm jack, which is the preferred method for the smart mirror.

Figure 2: Souncore 3 Speaker

In order to implement this speaker into the main assembly, it was necessary to disassemble it into its PCB and speakers. To add on, more speakers can be attached to the PCB with some soldering. After doing so it was placed at the bottom of the main assembly to prevent any blockage from the ventilation

Microphone

For this smart mirror project, the microphone was not a priority; thus the inexpensive JOUNIVO USB microphone was selected

Figure 3: USB microphone

Although the microphone was not prioritized, this microphone allows for 360-degree motion, with high sensitivity detection at various locations. This microphone allows for user privacy by including a mute button, and an indicator LED letting the user know with their voice is being heard. This microphone also includes noise-canceling technology, which helps prevent any unwanted noise from being picked up; however, one issue that arises is that the user's voice might become muffled and distorted. And similar to the IR frame, described below, this microphone is a plug-and-play device and requires no additional drivers. This microphone allows for more flexibility when creating the smart mirrors' main assembly.

Camera

- The team wanted to pursue the use of an Xbox Kinect for gesture recognition and control, however, due to technical challenges with the operating system this option was not pursued. The error occurred while trying to receive data from the Kinect but the device was not responding or visible to the Raspberry Pi.

- A Raspberry Pi camera module was then used to test features; however, the low resolution is not viable for the security camera application the team intends to use.
- A USB webcam was used due to its higher resolution and compatibility with The Android operating system.

Infrared Frame

Figure 4: IR frame

Infrared frames behave as touch screen devices, but are connected like a regular PC mouse. It has ultra-low power consumption so there is no need to worry about a large power supply. It is able to connect to various devices such as Windows, MAC OS, Linux, and even Android, which is the one being used for this project. It is important for the device it's controlling to have a USB port or an adaptor of sorts. This IR frame will work even when 25% of the sensors malfunction and has an operating temperature of -20 to 70C, so it's perfect for a home setting. And although this device should remain dry at all times, it is able to support a humidity of up to 90%.

Enclosure

Figure 5: Enclosure's 3D model

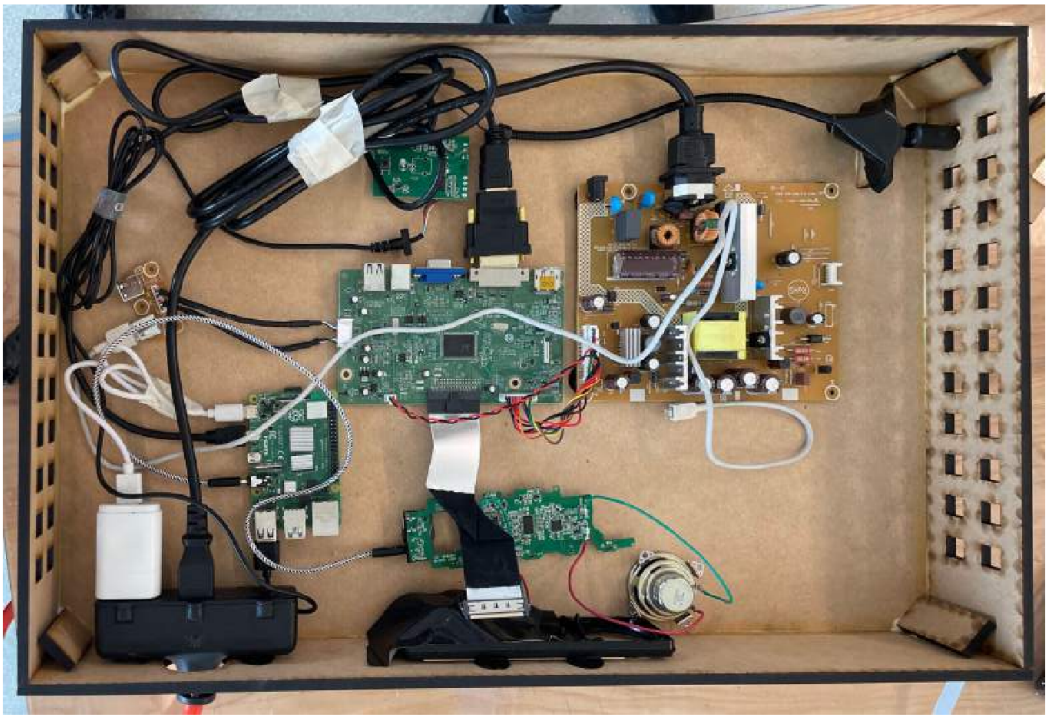


Figure 6: Interior Assembly

The team wanted to create a compact enclosure while maintaining a relatively low cost. In order to meet these goals, a design was created using Fusion 360, a 3D modeling software. These 3D files were then converted to their DWG counterpart, thus

allowing them to be laser cut onto an MDF board. This material is low in weight, durable, easily customizable, and quick to assemble. If modifications need to be made on the spot, MDF can be easily cut and glued back together.

Software

User Interaction

Operating System

In order to build a user interface, we first used Magic Mirror (MM) which was based on Linux. MM essentially allowed a pre-scripted GUI to run using Node.js and could be customized using modules to tailor it to our specifications. This interface was later changed for Android OS because of the difficulties in modules duration of support and its accessibility. The Android OS included an easy user interface with a large variety of software integrations. The Android OS included all the functionality of the Raspberry Pi and easy modifications for future design processes.

The base features of the Android OS in Raspbian are:

- Addition of physical buttons for power and volumes through the GPIO pins.
- Have SSH and VNC activated on start which can be used for remote access.
- Bluetooth and Tethering options for devices to connect with the Pi.

However, because the Android OS was implemented later, the complete functionality hasn't been tested and only used to implement the key features and goals for this project.

The user interface is easily accessible and modified according to different users.

Widgets

The Home screen primarily has the widgets that were the goals for this project. Our group started to look over and work on creating custom widgets which could interact with the user for monitoring the existing conditions of smart switches and security cameras. Along with showing the preview of pictures/ videos taken while the user was away from home. The widgets currently included in our smart mirror are:

- Weather and Time

- Spotify
- Alexa
- Google Assistant
- Calendar
- Warden Cam Security

The widgets are placed for easy access with room for additional widgets.

Apps

The applications were installed through Google Play Store which though similar to an Android tablet has the versatility and software updates which can be automatically installed without manual commands. The applications currently installed are Alexa, Google Services, music, and Warden Cam Security. The Alexa and Google voice assistants are applications used for the voice assistants of the mirror. For entertainment, we have installed Spotify, Apple, Amazon, Google music, and Youtube. These applications can be controlled by both user's touch input or through voice commands. The Calendar app displays the tasks and events of the week along with holidays, meetings, and reminders. The Warden Cam Security is an application that is used for monitoring and security purposes. With further work in Android Studio, we can integrate the Warden app to start and run in the background when the user is away from home.

Voice Assistants

Commands

Since the system is booted with Android OS, it has enabled us to integrate both Google and Alexa Voice services. The commands can be activated by clicking on a button. This is done with the intention to not activate voice commands unknowingly. We can create and use custom commands through Alexa and IFTT to control other smart devices and using the Play Store we can install and update the applications. This helps for easy user interaction.

News

The news is viewed via Google services and is automatically updated to the location of the nearest city through the internet. The user can invoke news both via Alexa and Google Assistant. There also is daily news which is displayed on the Home screen within the Calendar. Custom routines can be generated at specific times of the day to invoke the smart mirror to read out the news.

Technical Challenges

Raspian's Magic Mirror

Voice Services

Figure 7: Google voice assistant

As of September of 2021, Alexa Voice Services had stopped supporting custom builds on Linux. The sample build can only be invoked by manually calling it. Along with Google assistant having minimum support as of December 2021. Another technical challenge that took place was the voice assistant modules for Magic Mirror had limited support, all the modules were open-sourced, and difficult to set and manage for custom builds.

Kinect Gesture Control

Linux

While the implementation of OpenCV seemed promising with the utility that the Xbox Kinect offered due to the various sensors, we encountered a major setback. After countless hours of installing the proper directories and files, as testing proceeded it became clear that there were gonna be many issues with the Kinect. While the raspberry pi did recognize the device, and displayed information about it in the terminal, we were unable to retrieve the video that the XBox Kinect was capturing. We were able to get the device to turn on as well, however the graphics drivers that Raspbian uses, more specifically OpenGL, were not compatible and as a result, we were unable to test the kinect and retrieve the images. The image below shows the error we encountered numerous times while troubleshooting and trying different methods with the kinect.

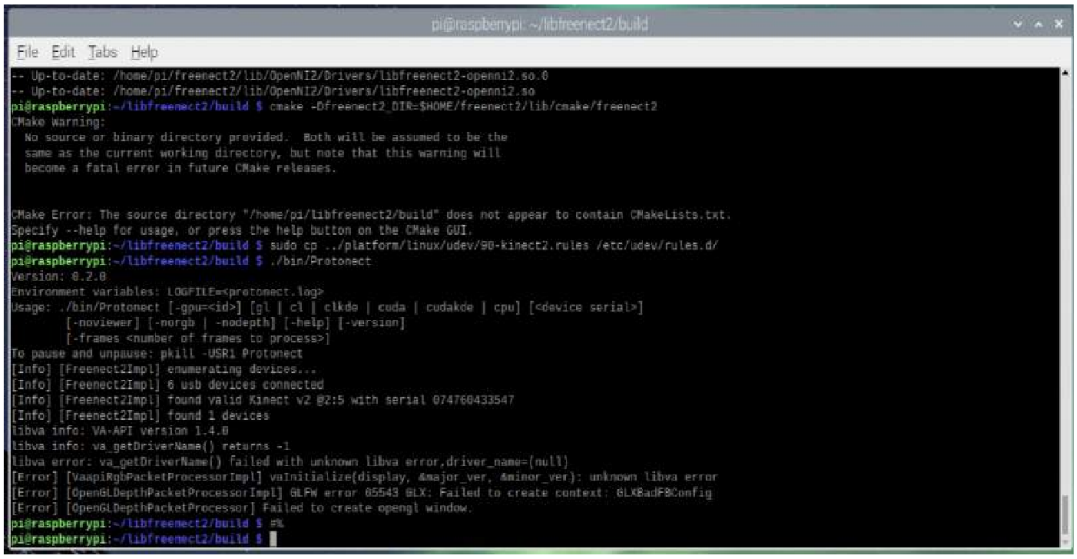


Figure 8: XBox Kinect Error

Broken Hardware

Monitor #1

In order to save space in our final product, the housing of the displays needed to be stripped. We used only the display panel and reorganized the various boards the monitor had inside the custom housing we created. During this assembly process, after properly connecting all the devices and placing everything within the housing, the

monitor would not turn on. After removing the monitor and after further inspection, we noticed the ribbon cable slot broke off the monitor itself, and as a result, we were unable to create the connection in order to give the monitor power. We believe it occurred due to some tension created within the housing due to some tight connections or spacing between components. Below is the image of the damage that occurred.

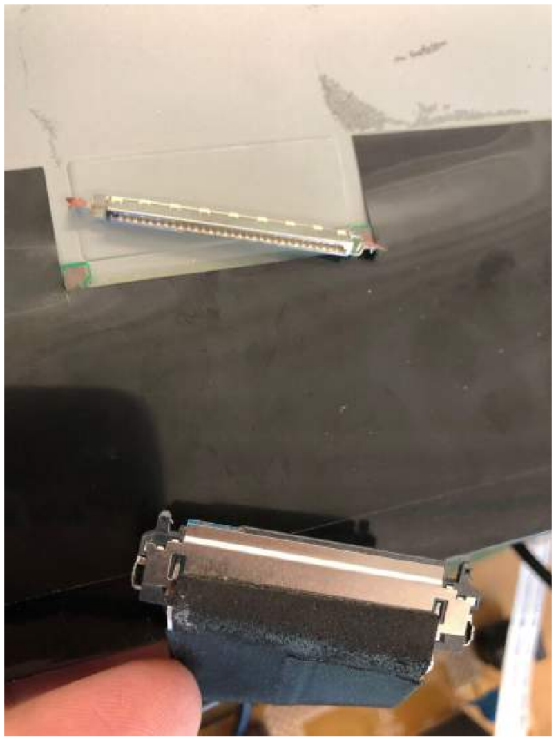


Figure 9: Broken monitor's display cable and connection

Monitor #2

Our second attempt was with an extra personal monitor we had available. The monitor was fully functional and we successfully stripped the housing and had it ready for assembly. However, before continuing, we began to test the functionality further. We encountered an issue with the video connections. The monitor displayed a message on the screen that said the frequency was not compatible with the device. We tried various connections and we tried changing the resolution and nothing resolved our issue. Ultimately, we proceeded to use an old TV as our display.

Compatibility Issue

In regards to changing the resolution on the display as mentioned under “Monitor #2”, we encountered some issues with the OS and display. At first, the resolution was not to scale and needed to be readjusted. However, when we proceeded to change the system resolution in the Raspberry Pi settings, changing the resolution resulted in an unusable device as the image would not be displayed despite the system running. This happened on various occasions as we tested multiple monitors and needed to change the resolution as well for previous tests. After the various occurrences, a complete reinstall of the OS had to be done in order to get the system running properly again.

Milestones and Interactions

Milestone		Tasks	Assigned To	Date
0 - Planning				
	0.1	Component Research	Everyone	
	0.2	Magic Mirror Research	Alan Palayil	
	0.3	Alexa API and Google Assitant Res	Alan Palayil	
	0.4	OpenCV Research	Gabriel Gutierrez & Fabian Garcia	
	0.5	Gesture Recognition Research	Gabriel Gutierrez & Fabian Garcia	
1 - Design - 01				
	1.1	Design Schematic	Everyone	
	1.2	Acquire Required Parts	Everyone	3/1
	1.3	Test Parts and Ensure Functionality	Everyone	3/1
	1.4	Alexa API and Google Assistant Im	Alan Palayil	3/10
	1.5	OpenCV Setup	Fabian Garcia &	3/10
	1.6	Magic Mirror Screen Setup	Alan Palayil	3/12
	1.7	Gesture Control Hand Mapping	Fabian Garcia	3/14
	1.8	Record Specific Gestures	Gabriel Gutierre:	3/21
	1.9	Gesture Implementation	Gabriel Gutierre:	3/21
	1.1	Speaker and Camera Connection	Everyone	3/28
1 - Design - 02				
	1.1	Implement android OS	Alan	4/1
	1.2	Test widgets	Fabian	4/1
	1.3	Look into security apps	Gabriel	4/1
	1.4	Compile Everything	Everyone	3/12
2 - Completeion				
	2.1	Create frame	Alan Palayil	4/15
	2.2	Assemble Components	Everyone	4/25

Table 1: Milestones and timeline

Broader Impact of the project

This mirror would be limited to a specific group of people who would like to add some more entertainment to their life and enjoy testing new products. This interactive smart mirror has the ability to play music, access various voice assistants, view the current time and weather, and much more. This product may also be more financially viable than its competitors so if it were released into the market, it could do pretty well; however, its own software would first have to be developed to avoid any copyright issues when having it on sale. A plus side to running Android OS, is the versatility the OS offers. Whether through customization of the screen or creation of Android apps. This versatility allows the system to be open to the user's interpretation.

Competitors:

- Vilros Magic Mirror V3 ($199)
  https://www.microcenter.com/product/635314/Magic_Mirror_V3_-_2_Way_Mirror_for_Smart_Mirror_Project_Includes_Internal_Ready_to_Connect_LCD?storeID=025

  Frameless Rectangular Anti-Fog Bathroom Vanity Mirror ($547)
- https://www.homedepot.com/p/ad-notam-Smart-24-in-W-x-30-in-H-Frameless-Rectangular-Anti-Fog-Bathroom-Vanity-Mirror-in-Silver-Neutral-60-CR7HO-NL/307397336?source=shoppingads&locale=en-UShttps://www.homedepot.com/p/ad-notam-Smart-24-in-W-x-30-in-H-Frameless-Rectangular-Anti-Fog-Bathroom-Vanity-Mirror-in-Silver-Neutral-60-CR7HO-NL/307397336?source=shoppingads&locale=en-US

As seen above, the price of a smart mirror can be a couple of hundred dollars different from one to another. The Vilros smart mirror seems to be the most cost-effective mirror, but the screen with all the information covers only a small portion of the mirror itself. If the team's smart mirror were to be placed on the market, the cost would probably be reduced when mass producing it.

Future potential

Although our current device seems like an intuitive smart mirror that can be displayed anywhere in your home, there are always improvements that can be done with both the assembly and software. First of all, the assembly can be made from a long-lasting material such as wood to ensure the mirror is both durable and decorative. Next, the 2-way mirror was made from acrylic, which presented some issues since the reflective paint can be easily scratched. This can be fixed by incorporating the more expensive two-way glass mirror. To add on, a custom speaker could have been created to provide the speaker with a more pronounced, high-quality sound.

For the future of this project on the software, the aspect is to have a target audience for Home Assistant and Smart Trainer. In both the modules, the device needs to have a custom monitoring system that can run in the background and not take up the whole display. The monitoring system can be attached to a motion sensor to notify the user via the app of movement when the user is away from home. The monitoring systems have a larger audience for babysitting and home security. In the Home Assistant, we can develop a virtual assistant that can look over the house similar to JARVIS from Iron Man and the appearance of the mirror makes it sleek and hidden. For the smart trainer, we have plans to develop a custom application that has computer vision capabilities to track the user while they are working out. The trainer can also track your workouts, create custom workouts, and track calories, and food intake. The OS itself can also be improved with more integration with sensors and apps.

Mass-producing the device may seem like a challenge and expensive to do, however, we believe the device is already cheaper than those on the market. Some additional money can be saved by creating our own custom IR frames, however, through some bulk buying of the necessary components, the price of the device per unit can be reduced. The frames themselves can be laser cut and assembled by hand and the remaining components would need to be acquired and connected properly. As for the

OS for each device, they would all be running Android OS by default, and a micro SD card will be included with the device.

Conclusion

While other electronic devices may remove you from your daily routine and serve as a distraction, the Smart Mirror is meant to be incorporated into your daily life and helps assist you when needed. It not only offers valuable information as the day progresses, but it gives you full control over what's displayed and how you want to interact with the device. It serves as a tool that makes getting ready for your day simpler and much more efficient while also making the preparation fun and interactive. To allow for a better-looking product, a relatively slim enclosure was created. With the help of Google's voice the current news, weather and time can be displayed in an instant. Although challenges arose in this project, the next viable solution was chosen to ensure the device kept working as intended.

Materials and Budget

Name	Price	Link	Quantity	Total Amount	Actual Amount
Pi 4 4GB Extreme Kit - 128GB	$149.95	https://www.canakit.com/raspberry-pi-4-4gb.html	1	$149.95	$149.95
Addressable RGB Strip 16.4Ft	$18.99	https://www.amazon.com/gp/product/B088BB8WTZ/ref=ox_sc_saved_title_2?smid=A35UAT07QG3EC6&psc=1	1	$18.99	
24V 3A DC Power Supply	$14.99	https://www.amazon.com/dp/B07Q29CD7J/ref=cm_sw_r_apan_glt_i_7MFQKT9W1W55A47DFJVF	1	$14.99	$14.99

DC Step-Down Converter	$13.99	https://www.amazon.com/DROK-Adjustable-Converter-Transformer-Protective/dp/B07JZ2GQJF/ref=sr_1_6?crid=1BK77CV0ZA57M&keywords=DC+Step-Down+Converter+%2824+-%3E5V+3A%29+%28link%29&qid=1644286298&s=electronics&sprefix=dc+step-down+converter+24+-+5v+3a+link+%2Celectronics%2C59&sr=1-6	1	$13.99	$13.99
Speakers	$49.99	https://www.amazon.com/Soundcore-Bluetooth-Diaphragm-Technology-Waterproof/dp/B08BCHKY52	1	$49.99	$49.99
Microphone	$15.99	https://www.amazon.com/Microphone-Condenser-Indicator-Gooseneck-Recording/dp/B07N2WRHMY/ref=sr_1_6?keywords=usb+microphone&qid=1644286906&sprefix=usb+microp%2Caps%2C80&sr=8-6	1	$15.99	$15.99
3.5mm cable	$4.99	https://www.amazon.com/CableCreation-Auxiliary-Compatible-Headphones-iPhones/dp/B01K3WX4FW/ref=pd_lpo_1?pd_rd_i=B01K3WX4FW&th=1	1	$4.99	$4.99
XBox One Kinect	$39.99	https://www.gamestop.com/gaming-accessories/controllers/xbox-one/products/microsoft-kinect-for-xbox-one/102581.html?gclid=Cj0KCQiA3fiPBhCCARIsAFQ8QzXYEH2r	1	$39.99	

		e1BVw5nMAqDBVqxpR9S-05hy4om79O2z1N74s-H0ePArvjEaAnSFEALw_wcB&gclsrc=aw.ds			
Kinect Adapter	$21.99	https://www.amazon.com/Adapter-Xbox-One-Windows-Certified-Updated/dp/B07JHJM9FG/ref=sr_1_3?keywords=kinect+adapter&qid=1644089351&sprefix=kinect+%2Caps%2C88&sr=8-3	1	$21.99	$21.99
24" Two-Way Acrylic Sheet	$36.99	https://www.amazon.com/0-04-Acrylic-See-Through-Mirror-Transparent/dp/B01CZ35XWY?th=1	1	$36.99	$36.99
24" IR Frame	$155.00	https://www.amazon.com/Multi-Touch-Point-Infrared-Screen-Overlay/dp/B07CW9WWT5/ref=sr_1_3?crid=4BLQ8682VDNQ&keywords=IR+Frame+24+inch&qid=1644529117&sprefix=ir+frame+24+inch%2Caps%2C93&sr=8-3	1	$155.00	$155.00
3-Outlet Power Extension Cord	$15.99	https://www.amazon.com/KMC-3-Outlet-Extension-2-Pack-1875Watt/dp/B07BLQ5KN7/ref=sr_1_5?crid=QO5FVDEX401S&keywords=2%2Boutlet%2Bpower%2Bextension&qid=1645905418&s=electronics&sprefix=2%2Boutlet%2Bpower%2Bextension%2Celectronics%2C72&sr=1-5&th=1	1	$15.99	$15.99
			Total Price	$538.85	$479.87

Table 2:Total spent on materials

User Manual

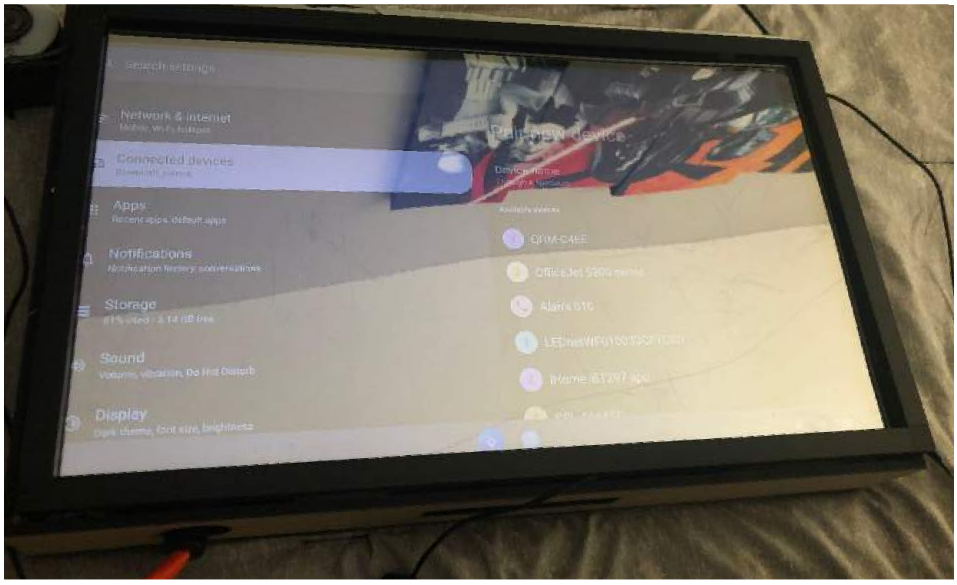


Figure 10: Smart mirror final product

Basic Startup

1. Properly connect devices to a power source and ensure all connections are fully seated.
2. Once connections are double-checked, proceed by clicking the power button on the right side of the display, near the top right corner. Once pressed, the boot process will begin and Android OS will begin to boot.
3. Once the device is up and running, you are free to do as you please.

Spotify:

1. To use Spotify services, locate the Spotify app on the home screen, or the Spotify widget and tap on the icon. Spotify will begin to launch.
2. Once fully loaded, you can search or browse music.
3. Once playing, the music controls can be accessed through the home screen widget, or through Alexa and Google voice assistants

Alexa:

1. To activate Alexa, locate the Alexa icon on the home screen and tap on the icon.

2. An Alexa window will open and you will be prompted to speak. Say what you would like to ask.

Google Assistant:

1. Similar to Alexa, google functions in the same way. Tap on the Google Assistant icon.
2. Wait for the window to open, and begin to ask what you need to do.

Google Calendars:

1. Click on the widget displayed on the home screen.
2. Once in the calendar menu, the user can create custom events and update their google calendar from the Smart Mirror.

Miscellaneous Applications:

1. Similarly to the above mentioned apps, all the applications can be accessed through the various icons.
2. In order to access the remaining apps that are not present on the home screen, swipe up on the display and the applications will be displayed.
3. Select the application you would like to use.

Display and Android Setting:

1. In order to access the android and display settings, swipe down on the display and a drop down menu will appear.
2. Begin by modifying the necessary settings and swipe back up to close the drop down.

Individual Contributions

Member	Contributions
Alan	1. The operating system (Magic Mirror and Android OS) researched and tested the operating system for the

	project. a. Magic Mirror is an open-source GUI for Linux. b. Android OS based on Raspberry Pi with which you can still use Raspberry Pi"s GPIO pins and services. 2. Software integration (Spotify, AVS, Google Assistant) researched and implemented various applications to run with the OS to achieve the desired goals. a. Integrated the voice assistant apps: Alexa and Google Assistant. b. Implement Spotify through RaSpotify (Magic Mirror) and Spotify Android through Play Store.
Fabian	1. Gesture Control through Kinect and Camera module a. Further research and troubleshooting with the Kinect. b. Testing gesture control with a Camera Module prior to changing Operating Systems. 2. Construction a. Helped assemble the final product 3. Android widgets and Future App Development a. Working on various widget implementations as well as the possibility of our own apps for future use. Time is the limiting factor.
Gabriel	1. OpenCV for Facial Recognition a. When using the original operating system, a photo booth program was tested for raspbian b. Assisted in installing the Xbox 2. Security app a. Research different security webcam apps that were compatible with the Android OS operating system b. Tried to get the Google Drive cloud save feature working 3. Enclosure a. Designed the enclosure using Fusion 360 b. Help assemble the final enclosure.

References

1. https://developer.amazon.com/en-US/docs/alexa/avs-device-sdk/raspberry-pi.html
2. https://pimylifeup.com/raspberry-pi-google-assistant/
3. https://howchoo.com/project/mzu3ndm2otu/building-a-voice-controlled-smart-mirror-with-raspberry-pi-and-jasper
4. https://ei23.com/diy-smarthome/
5. https://www.hackster.io/yogai/yogai-smart-personal-trainer-f53744
6. https://github.com/MichMich/MagicMirror/wiki/3rd-Party-Modules
7. https://docs.magicmirror.builders
8. https://konstakang.com/devices/rpi4/LineageOS19/
9. https://opencv.org/
10. https://pimylifeup.com/raspberry-pi-opencv/
11. https://github.com/smellslikeml/ActionAI
12. https://github.com/abhiTronix/OpenCV_Raspberry_pi_TBB
13. https://www.raspberrypi.com/news/all-seeing-pi-photo-booth/

Appendix

Android 12

lineage-19.1-20220407-UNOFFICIAL-KonstaKANG-rpi4.zip found in

https://konstakang.com/devices/rpi4/LineageOS19/

is the Android 12 operating system for the Raspberry Pi

Photobooth program

```
import numpy as np
import cv2

face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

choice = int(input("Enter filter:"))



specs_ori = cv2.imread('glass.png', -1)
cigar_ori = cv2.imread('cigar.png', -1)
mus_ori = cv2.imread('mustache.png', -1)
dog_ori = cv2.imread('mustache.png', -1)
nerd_ori = cv2.imread('mustache.png', -1)
rab_ori = cv2.imread('mustache.png', -1)


# Camera Init
cap = cv2.VideoCapture(0)
cap.set(cv2.CAP_PROP_FPS, 30)


def transparentOverlay(src, overlay, pos=(0, 0), scale=1):
    overlay = cv2.resize(overlay, (0, 0), fx=scale, fy=scale)
    h, w, _ = overlay.shape  # Size of foreground
    rows, cols, _ = src.shape  # Size of background Image
    y, x = pos[0], pos[1]  # Position of foreground/overlay image

    for i in range(h):
        for j in range(w):
            if x + i >= rows or y + j >= cols:
                continue
            alpha = float(overlay[i][j][3] / 255.0)  # read the alpha channel
            src[x + i][y + j] = alpha * overlay[i][j][:3] + (1 - alpha) * src[x + i][y + j]
```

```
    return src

while 1:
    ret, img = cap.read()
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(img, 1.2, 5, 0, (120, 120), (350, 350))

          for (x, y, w, h) in faces:
        if h > 0 and w > 0:
        if choice = 1
            glass_symin = int(y + 1.5 * h / 5)
            glass_symax = int(y + 2.5 * h / 5)
            sh_glass = glass_symax - glass_symin

            cigar_symin = int(y + 4 * h / 6)
            cigar_symax = int(y + 5.5 * h / 6)
            sh_cigar = cigar_symax - cigar_symin

            mus_symin = int(y + 3.5 * h / 6)
            mus_symax = int(y + 5 * h / 6)
            sh_mus = mus_symax - mus_symin

            face_glass_roi_color = img[glass_symin:glass_symax, x:x + w]
            face_cigar_roi_color = img[cigar_symin:cigar_symax, x:x + w]
            face_mus_roi_color = img[mus_symin:mus_symax, x:x + w]

            specs = cv2.resize(specs_ori, (w, sh_glass), interpolation=cv2.INTER_CUBIC)
            cigar = cv2.resize(cigar_ori, (w, sh_cigar), interpolation=cv2.INTER_CUBIC)
            mustache = cv2.resize(mus_ori, (w, sh_mus),
interpolation=cv2.INTER_CUBIC)

            transparentOverlay(face_glass_roi_color, specs)
            transparentOverlay(face_cigar_roi_color, cigar, (int(w / 2), int(sh_cigar / 2)))
            transparentOverlay(face_mus_roi_color, mustache)
        elif choice = 2
            dog_symin = int(y + 1.5 * h / 5)
            dog_symax = int(y + 2.5 * h / 5)
            sh_dog = dog_symax - dog_symin
            face_dog_roi_color = img[dog_symin:dog_symax, x:x + w]
            dog2 = cv2.resize(dog_ori, (w, sh_dog), interpolation=cv2.INTER_CUBIC)
            transparentOverlay(face_glass_roi_color, dog2)

        elif choice = 3
            nerd_symin = int(y + 1.5 * h / 5)
            nerd_symax = int(y + 2.5 * h / 5)
            sh_nerd = nerd_symax - nerd_symin
```

```
            face_nerd_roi_color = img[nerd_symin:nerd_symax, x:x + w]
            nerd2 = cv2.resize(nerd_ori, (w, sh_nerd), interpolation=cv2.INTER_CUBIC)
            transparentOverlay(face_nerd_roi_color, nerd2)

        else choice = 4
            rab_symin = int(y + 1.5 * h / 5)
            rab_symax = int(y + 2.5 * h / 5)
            sh_rab = rab_symax - rab_symin
            face_rab_roi_color = img[rab_symin:rab_symax, x:x + w]
            rab2 = cv2.resize(rab_ori, (w, sh_rab), interpolation=cv2.INTER_CUBIC)
            transparentOverlay(face_glass_roi_color, rab2)

            break


    cv2.imshow('Thug Life', img)
    key = cv2.waitKey(1) & 0xFF
    if key == ord("q"):
        break

    k = cv2.waitKey(30) & 0xff
    if k == 27:
        cv2.imwrite('img.jpg', img)
        break

cap.release()
cv2.destroyAllWindows()
```