RecognEyes

Authors: Mary Day, Angelo Pagliuca, & Aria Paradkar. Electrical and Computer Engineering, Carnegie Mellon University

***Abstract* — A particularly difficult stage of dementia is when the individual no longer recognizes their family members' or others' faces. Thus, we decided to create RecognEyes. RecognEyes is a pair of glasses for the individual with dementia with a small camera on the lens and a small speaker by the ear. We implemented facial recognition to detect the person with whom the diagnosed individual is interacting (interactor), and say their name through the speaker. RecognEyes is also the name of our deployed web application where anyone can be entered in the database, by the user or perhaps by the user's aid.**

***Index Terms* — Bone Conductor Speaker, Camera, Dementia, Design, Facial Recognition, Interactor, Localization, OpenCV, Raspberry Pi, Trained Model**

I. INTRODUCTION

Alzheimer's disease is a type of dementia that slowly destroys an individual's memory and thinking skills. Unfortunately, there is not yet a cure for the disease but medical advances and research have allowed individuals with dementia to live much longer after diagnosis. A particularly difficult stage of dementia is when the individual no longer recognizes their family members' or others' faces. Since our group members have had family members who have been affected by the disease, we all jumped at the opportunity to help someone with any type of dementia, specifically Alzheimer's disease. RecognEyes will help individuals diagnosed with the disease by using facial recognition to determine and tell the diagnosed individual who they are interacting with. Currently, there are no devices that do this. A major advantage of our approach is that it is easy to implement into one's day to day life by simply charging and wearing the glasses and using the web application to add new users into the database.

Our primary goal was to have the speaker correctly determine who the user is interacting with via facial recognition with at least 95% accuracy in 3 seconds or less. We achieved the accuracy goal but not the timing goal; we can recognize a person in around 6 seconds. We achieved a false positive rate of about 28% and a false negative rate of about 7%. For the speaker, we expect a similar accuracy, false positive rate, and false negative rate as the facial recognition. Other goals we have are user comfort and easy integration into one's day to day life. We have a user interface that is easy to navigate and add new people into the database. We made the apparatus as small as possible and comfortable to wear. The battery lasts at least 6 hours.

II. DESIGN REQUIREMENTS

We chose the requirement of 95% accuracy for people in our database because many papers we read [1] on facial recognition have accuracies in the 95-99% range. We figured that since we are using a simpler algorithm, a less computationally intensive CPU, and we have fewer people in our database, our accuracy will be slightly less than the ones in the papers. We initially tested this by having at least three people in our database to start, which would be the three people on our team, and seeing how well the algorithm does at recognizing each of us, then we plan on retraining the model with more people in the database, and seeing how it does on more people. Our training set now has our three group members as well as the main characters from Jurassic Park. We have 3-6 images of each person in our testing set. This includes a few images with multiple people in it, both in and not in our training set. We also included around 5-8 images of people not in our training set to test how well classification of unknown people works. We tested our model on these images of people not in the database, as well as on Mary's sister who looks a lot like her but is also not in the database. We attained a false positive rate of 28%, which is better than we expected, and a false negative rate of 7%, which is also better than we expected. We chose a false positive rate limit of 30% because we take pictures approximately once every six seconds and since we are training with so few images per person, there is bound to be at least one orientation a person could be in such that the recognition algorithm is fooled, and thinks the person is someone in our database when they are actually not. We chose a false rejection rate of 15% because we feel that having one in every seven pictures taken of someone in the database be inaccurate is reasonable since we will be taking many pictures. We prioritize a low false negative rate over a low

false positive rate since it is better for our use case to mistake a person we do not actually know for someone we do know, than to not recognize somebody we should know. We want to find a balance between minimizing false negative rate and the time it takes to recognize a person, while maximizing the accuracy of the algorithm. Accuracy of the facial recognition is the most important requirement for our project since our device would be more useful if it took more time to say the correct name as opposed to saying an incorrect name quickly because they would simply become more confused.

We chose the requirement of having the name spoken in less than three seconds because that is the longest amount of time we felt was appropriate for a human to recognize someone they know without it being too long or becoming overly uncomfortable. However, we felt that if this time requirement is straining our accuracy too much, it was the best decision to increase our time requirement rather than to sacrifice the accuracy, for the reasons described above. This time requirement of three seconds encompasses the path from facial recognition to raspberry pi to speaker to ear. We have broken this down into a time requirement for each segment of the path: for the facial recognition to identify the person or people in each picture, for the label to travel through the Raspberry Pi, and for the speaker to receive the label and say the name. Our final product took approximately six seconds to achieve this goal.

The camera will take a picture every six seconds. It takes less than half a second to take a picture when prompted. We originally planned on two pictures every second, but the algorithm took a bit longer than planned to perform the facial recognition algorithm. We chose to take images at the rate in which they are processed because otherwise we would be unnecessarily taking pictures and not using them. We expect our users to be interacting with individuals for longer than six seconds, so we do not expect any individuals to go unnoticed.

Once the camera takes its pictures, the Raspberry Pi processes the image through facial recognition; i.e. the model receives the image, crops out the faces, converts it to 128-bit vectors, checks the database of saved 128-bit vectors of known people, and labels the image. We chose this because the base facial recognition API we are using works for the Raspberry Pi and does not use the most advanced methods in the field. Thus, we predict that it will take longer than the average in the industry, which is around 0.4 seconds. We feel that five times the average is a good maximum time requirement.

Our speaker audibly says the name and relationship of the interactor by using Espeak. This is accomplished in under one second. We were originally going to use Amazon Polly because we were already going to be using AWS and we thought it would be easier to integrate the whole system if we used products by the same company. We instead found that it was much easier to integrate the python module Espeak since there was a python command to say words and we had to integrate it with our python script.

These portions together add to be approximately six seconds. This is five seconds longer than our original plan in order to ensure our accuracy can reach at least 95%.

We chose the Arducam Sensor Spy Camera Module because it is made for the Raspberry Pi 4 [3], so it was very easy to integrate with the overall system. We minimized the size of the camera since we put the camera directly on the glasses' lens' and we do not want to obstruct the user's view or feeling of balance on both sides of the glasses. Our camera is 0.25 inches squared and we could not find one smaller than this. Even while minimizing the size, we maintained a high enough resolution to be able to carry out facial recognition. Since OpenCV requires a 96x96 pixel image of just the face, our camera must be good enough to take a picture of that size, so we needed our resolution to be at least 720p. Our camera is 1080p while taking pictures, which is perfectly in line with the facial recognition requirements. We previously planned to use a different camera by the same company, but that camera was 0.5 inches since the camera circuit was on the back of the camera itself. We planned on rebuilding that circuit into the frame of the glasses and attaching the camera onto the lens of the glasses since it would be too clunky otherwise, and wiring up the rest as is. Now that the circuit is separate from the camera, and thus the overall size is smaller, we no longer had to do that; we just wired an extension flex cable to the end of the camera circuit to connect it to the Raspberry Pi.

For the flex cable, we chose the longest one we could find that would still work for the components we already had, like the camera and the Raspberry Pi. The one we found was the A1 FFCs Black Flex Cable for Raspberry Pi Camera [4]. This is two feet long, which is twice as long as the cable that came with the camera. We feel that this would be a comfortable length for the user, since this wire runs down the front of the user and connects from their left ear to their left hip. We were originally going to have the wire run down the user's back from their ear to their hip so that the wire is not in the way, but this would require the flex cable to bend. The camera works better if the flex cable is straight, so we changed the position of the wire to run down the front instead, although this adds a bit of inconvenience. We chose the left side for the camera and Raspberry Pi since a greater percentage of the population is right-hand dominated. However, we can mirror the design to maximize comfort for left-hand dominated people as well.

To connect the camera to the extended flex cable, we needed a cable extender attachment. We chose the

Adafruit CSI or DSI Cable Extender Thingy for Raspberry Pi [5]. This was the cheapest one we found that had good reviews.

We chose to use a Raspberry Pi 4 [6] because it is small enough to fit in a carrying case on the hip without being too cumbersome, but it has a fast and powerful enough SoC to run our facial recognition algorithm. It also has various ports for power, including a USB-C port, an audio jack which was useful for our speaker unit, and it is compatible with the camera we found. The Raspberry Pi 4 is also easily integrable with the base facial recognition API we found. We considered using a Jetson Nano, but this was slightly larger than the RPi 4, twice as expensive, and was not very easily translatable to the Raspberry Pi in terms of ports and connections available.

We initially chose the speaker Bone Conductor Transducer with Wires - 8 Ohm 1 Watt [7] because it is very small and works well with the hearing impared. Since we expected our users to be in an older age range, and perhaps have a hearing aid, we chose this to maximize user comfort. The bone conducting transducer is attached to two wires, so it would have been simple to connect to the Raspberry Pi via an amplifier circuit and a wire to audio jack cable. We planned to create the audio amplifier circuit on the right side of the glasses while using a 3.5mm Male Plug to Bare Wire Open End TRS 3 Pole Stereo 1/8" [8].

We needed to use an audio amplifier with this speaker since the audio will likely not be loud enough from just the speaker and the Raspberry Pi alone. We planned to make this circuit from scratch and integrate it into the frame of the glasses, but for testing purposes, we bought an off-the-shelf amplifier Stereo 2.1W Class D Audio Amplifier - TPA2012 [9].

Unfortunately, due to the Coronavirus pandemic we transitioned to virtual learning from our respective homes. This created several issues with the integration of our speaker components due to lack of equipment, mainly a soldering iron. We tried using a breadboard for the amplifier circuit but unfortunately the connections would break often with any movement of the Raspberry Pi. After several trial and errors to get this speaker circuit fully functional, we decided to change this part of the project. With not much time to order a new speaker, as package delivery times are longer than usual, we decided to use what was readily available to us at the time which was the Altec Lansing Speaker.

As for the accuracy of the speaker, we expected the same accuracy as the facial recognition because Espeak can pronounce the labels we give it if they are composed of syllables in its lexicon set.

We chose the battery pack USB Battery Pack for Raspberry Pi [10] because it is about the size of the Raspberry Pi so it would fit easily in the carrying case, and it would provide enough power to make our apparatus last for around six hours. We chose this limit of six hours because that is the shortest amount of time a human could reasonably expect to be identifying people for in any given day. We had to find a balance between power consumption and lasting power, since running facial recognition and text-to-speech are power intensive but our device is not useful if the glasses must be plugged into a wall to use them.

We chose the requirement of 80% user friendliness because when searching for a product, people usually filter by 4 stars out of 5 or above, with the average for Amazon products being 4.4, corresponding to 88% likeability.

We also chose to keep the image size of each picture in the database limited to between 2KB and 1MB in order to reduce our database and keep it as efficient as possible while still being able to have good enough resolution for our facial recognition.

III. PRINCIPLE OF OPERATION

Our Principle of Operation consists of three parts: Web Application and Facial Recognition, Camera System, and Speaker System. These three systems combined will make the whole of RecognEyes. These three systems are described at a greater length in this section.

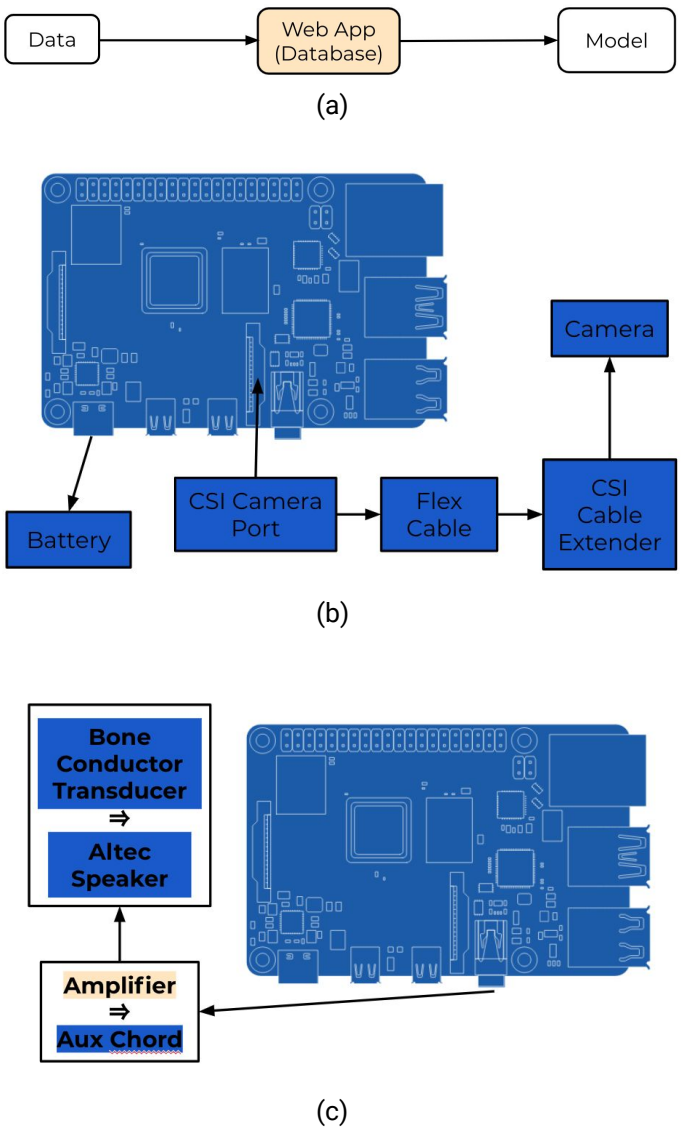


Fig. 1. System picture. (a) facial recognition system. (b) camera system. (c) speaker system.

The facial recognition system (Figure 1.a) consists of both the web application and the facial recognition learning model. We will use the Web Application as our main data collection system, as it is designed with user experience in mind -- in our case, caregivers of people diagnosed with dementia. Through the site, the user can make a profile where they can add pictures of their loved ones, or people that they will want to be recognized by RecognEyes. Once the user has enrolled their respective profile, we automatically use our training algorithm to first locate the face in each image using HoG, then create 128-d vectors out of them, and store those in an encodings file.

The camera system (Figure 1.b) consists of 2 main components: the flex cable, and the camera itself. These components are going to be tethered to the Raspberry Pi. Inside our main python script, the camera will take a picture every 6 seconds and send it to the Raspberry Pi through the CSI Camera Port where our algorithm is able to run the facial recognition. Our recognition algorithm inputs the image and the known encodings, locates the faces in the image using HoG, creates 128-d vectors out of each face, and compares each of these encoding vectors to those in our known encodings. It uses a Support Vector Machine and K-Nearest-Neighbors to determine the labels. Once it recognizes the interactor, it will let the user know through the speaker system. Once a person's name is recognized and spoken, the camera will once again take a picture and the operation cycle will continue.

The speaker system (Figure 1.c) only consists of an Altec Speaker connected to the audio jack via aux cord. After the interactors are recognized, the speaker will say the interactors' names and relationships to the user, if they are in the database. If an interactor is not in the database, the speaker will say "Unknown". The speaker will not repeat the interactor every time they are recognized, which is every 6 seconds. Instead, the speaker will only repeat the name and relationship of the interactor a maximum of twice every ten minutes.

IV. DESIGN TRADE STUDIES

A. Raspberry Pi 4

Prior to our proposal presentation, we had decided on the following hardware: a Raspberry Pi 4, a Bone Conduction Speaker, and an Arducam RPi Camera Model V2-8 megapixel. Only the Raspberry Pi 4 will be in our final product as the camera and speakers have been replaced. We chose the Raspberry Pi 4 primarily because it works well with facial recognition algorithms, specifically the Haar Cascade algorithm which we plan to use. In addition, the RPi4 is low cost, has variable RAM, and is USB powered by 5V and 2.5A.

B. Bone Conduction Speaker

We chose the bone conduction speaker because it works well for the hearing impaired who may or may not need a hearing aid. Since most individuals diagnosed with dementia tend to be 60 or older and need a hearing aid often, this speaker seemed to be the best option.

Unfortunately, due to the transition to virtual learning, the integration of this speaker into our final product posed several issues. Because of the lack of time left in the course to order a new speaker, we decided to use what was readily available to us, which was the Altec Lansing Speaker. This is connected to the Raspberry Pi via aux cord and works well with the final product.

C. Arducam RPi Camera

The camera was originally chosen due to its small size and high resolution, which would work well with facial recognition. With this camera, we expected a task of re-creating the circuit behind the camera to fit well on the final apparatus, which we expected to take at least 2 weeks. However, research on this task did not prove successful as we could not determine a proper way to detach the camera and reconnect to the new circuit, especially without having to buy new parts. Thus began the search for a new smaller camera with a high resolution. We finally found the Arducam ¼ inch 5 Megapixels Sensor Spy Camera Module. This camera seemed like an obvious choice for multiple reasons; not only was it a small camera without the circuit beneath it, but it also has a high resolution, and can improve our goal of user comfort with the overall apparatus. We now plan to attach the camera on the top left side of the glasses to the left hip,, where it will connect to the RPi4. In doing so, we will use a flex extension cable and a cable extender piece which allows us to connect two 15-pin 1.0 pitch cables end-to-end and extend the length.

D. *Raspberry Pi Facial Recognition Base API*

We chose the Base API mainly because it is compatible with the RPi4 and real-time facial detection. We wanted to use real-time facial detection because we would like RecognEyes to identify or verify an individual from a video frame. Since our user's frame is constantly changing, our camera will be taking photos continually. In addition, we planned to take approximately 15 photos per person and the base API we found aligns well, with a suggested minimum of 10-20 photos per person.

E. *Haar Cascade Algorithm*

The Haar Cascade Algorithm is a machine learning algorithm that is often used to detect and localize an object in images or videos. We chose this algorithm primarily because it works well with a Raspberry Pi 4. In addition, this is the facial recognition algorithm used in the base API we found above. Lastly, this algorithm is very fast which will contribute towards our goal of quickly performing the algorithm to detect who the user is interacting with and to tell the user who the individual is through the speaker.

V. SYSTEM DESCRIPTION

Our overall system diagram is provided at the end of the report. There we show how every subsystem is connected to each other. Here are the specific details for each subsystem in our system, going more in depth than our Principle of Operations (Section III).

A. Facial Recognition

The Facial Recognition subsystem is the main functionality of the apparatus. There are a couple parts that come together to make this subsystem work: the camera, an external computer, and the Raspberry Pi.

On an external computer, in a database hosted by AWS, we store many images of people we want to recognize. We store everyone who has an apparatus in this database, but we partition the database based on the user, so each model we train is based on the people in each account as opposed to having just one model we use for all people and all accounts. Using this reduced dataset, we detect, localize, and crop the face in each image using python's face_recognition module, built on DLib and OpenCv, to a 96x96 image. It also makes sure that the eyes and nose are in the same approximate place in each image, so the faces may appear distorted. Once it does this, it then passes the image to OpenCV to convert the processed image to a 128-bit vector, then trains a multi-class SVM using the 128-bit vectors we have collected. We upload this trained model to the Raspberry Pi automatically.

Using the camera, we take one image every six seconds from the live video feed, and use Haar-Cascades to detect, localize, and crop the image. We then convert it to a 128-bit vector and pass this to the model. After we do this, the model compares the vector to the database of known vectors and outputs a label.

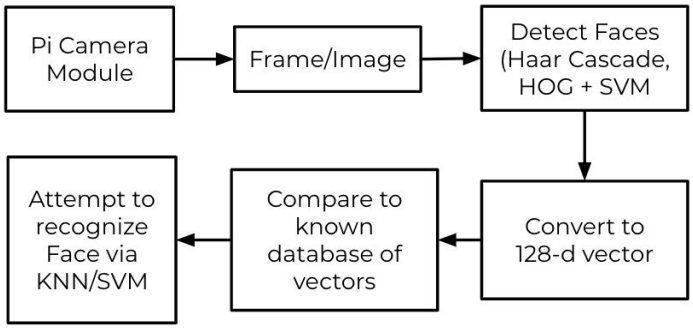


Fig. 2. Facial Recognition Diagram

B. Web Application

The Web Application subsystem is the one with the highest level of user experience, and thus is the one with the greatest human interaction. In our case, our main users will be people in older generations, and thus need a little more help with the website navigation. This is the reason the web application has been divided into five main sections: home page, creators section, design section, development blog page, and device page.

In order to make our user experience as smooth as possible, we based our design on the Snapchat Spectacles' website since we knew that this website is successful commercially, and the product is a pair of glasses, similar to our device. To diversify ourselves from the Snapchat Spectacles website and to focus more on our pool of users, we changed a few things, such as enlarging the font for easier readability, and displaying fewer vibrant colors.

The homepage will be the first site seen when exploring the application, and it is also where we first introduce RecognEyes. We introduce small aspects of the device to the user through small sections, where the user can go further into the page to get more information.

The device page will be where the user inputs their data. To access this page, the user needs to make an account. This allows us to keep the database reduced and efficient. Once the user is logged in, they will be allowed to add the names and relationships of the people they want to recognize into their profile. Once people are added to the profile, the user can add pictures between 2KB and 1MB to our database. After enrolling the people and pictures, the data will be downloaded onto the Raspberry Pi and our facial recognition algorithms will automatically train the model and start recognizing people.

The dev-blog page is included to help the viewer see the work accomplished in depth by each creator, how it was designed, in addition to looking at how the project progressed during the semester. These pages are included for the documentation of the creators, with no specific technical purpose.

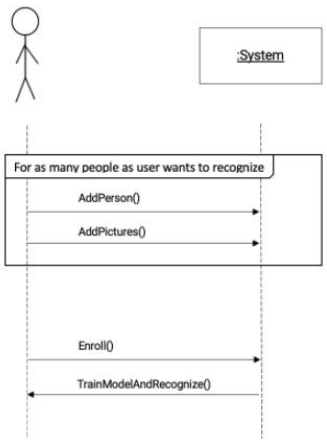


Fig. 3. System Sequence Diagram

C. Hardware

There are two main hardware components of our project that attach to the Raspberry Pi 4: the camera, and the speaker.

The camera system consists of the flex cable, the CSI cable extender, and the camera itself. The camera will be taking one photo every six seconds and will send the photos to the Raspberry Pi 4. OpenCV and the Haar Cascade algorithm are installed onto the RPi4 SD card and assist in facial recognition detection. The Raspberry Pi 4 runs the Haar Cascade algorithm using the photos taken on the RPi4 to identify if the interactor is in the database on the RPi or not. If the interactor is in the database, the speaker will tell the user the interactor's name and relationship to the user. If the interactor is not in the database, the speaker will tell the user that the individual is unknown.

We planned for the speaker system to consist of the bone conductor transducer connecting to an amplifier which would attach to the Raspberry Pi 4 through the audio jack. This speaker circuit would have been powered by the 5V pins on the RPi4. The amplifier would receive electrical signals from the RPi4 using a text-to-speech algorithm and then the amplifier would amplify those signals. These signals would be the words of the name and relationship of the interactor to the user. The amplifier circuit would have consisted of several polarized capacitors, several non-polarized capacitors, an amplifier chip, and the dual potentiometer. Refer to circuit diagram below. However, as mentioned earlier, we are no longer using the bone conductor transducer due to technical difficulties from the transition to virtual learning. Our Raspberry Pi is now connected to an Altec Lansing Speaker via aux.

The final product of our hardware components can be seen in Figure 4. A table consisting of our metrics, validations, and results can be seen in Figure 5.

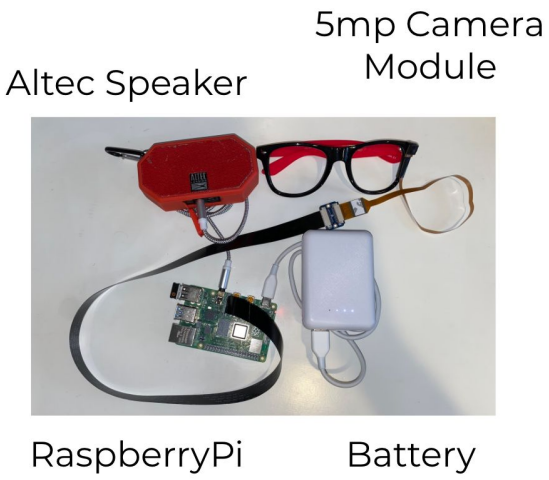


Fig. 4. Final Product

Feature	***Metrics***	***Validation***	***Results***
Facial Recognition	≥ 95% accuracy (in database), ≤ 3 seconds to recognize, ≤ 15% false rejection rate, ≤ 30% false positive rate	Python's time module, testing on team members, family members, and various celebrities	95% accuracy 6 seconds to recognize 7% false rejection rate 28% false positive rate
Speaker	Same accuracy as face recognition	Audible to user, Correct name and relationship spoken	Same accuracy as face recognition Slight British accent
Battery	≥ 6 hours	Different usages of all features	6 hours of battery life
Hardware User Comfort	≥ 80% ratings	People aged 18+ Google survey Family Members tested 15 people tested in total	80% rating
Software User Comfort	≥ 80% ratings	People aged 18+ Google survey Family Members tested 15 people tested in total	86% rating
Image Limit Size	≥ 2 KB ≤ 1 MB	Hard Software Limit Server Soft Memory Limit	Requirement stated on WebApp

Fig. 5. Metrics, Validation, and Results

VI. AWS CREDITS

We decided to use AWS because it was most easily available to us. We used the Amazon Elastic Compute Cloud (Amazon EC2) with an Ubuntu operating system as our desired base since it was well documented online for deployment. Since we used our free student account, we did not use any credits. Thank you so much Amazon. We love you.

VII. PROJECT MANAGEMENT

A. Team Member Responsibilities

We have decided to split our project into three main sections: Facial Recognition, Web Application, and Hardware. Within those three main sections we all have primary sections, Aria taking charge on facial recognition, Angelo creating the web application, and Mary assembling the hardware. In addition, we each had secondary roles between the sections. Aria and Angelo's secondary roles were to work on the hardware. Mary's secondary role was to help with both the facial recognition and the webapp.

B. Schedule

As we discussed in the above sub section, the project is divided into three main sections, with one person assigned to a main section. We did this in order to parallelize the work as the semester continued. Figure 5 represents our full schedule for completion of the project. We left a lot of slack to prepare for unforeseen problems. Throughout the semester, we did in fact encounter several problems, specifically the transition to virtual learning, but they did not have a tremendous impact on our scheduling.

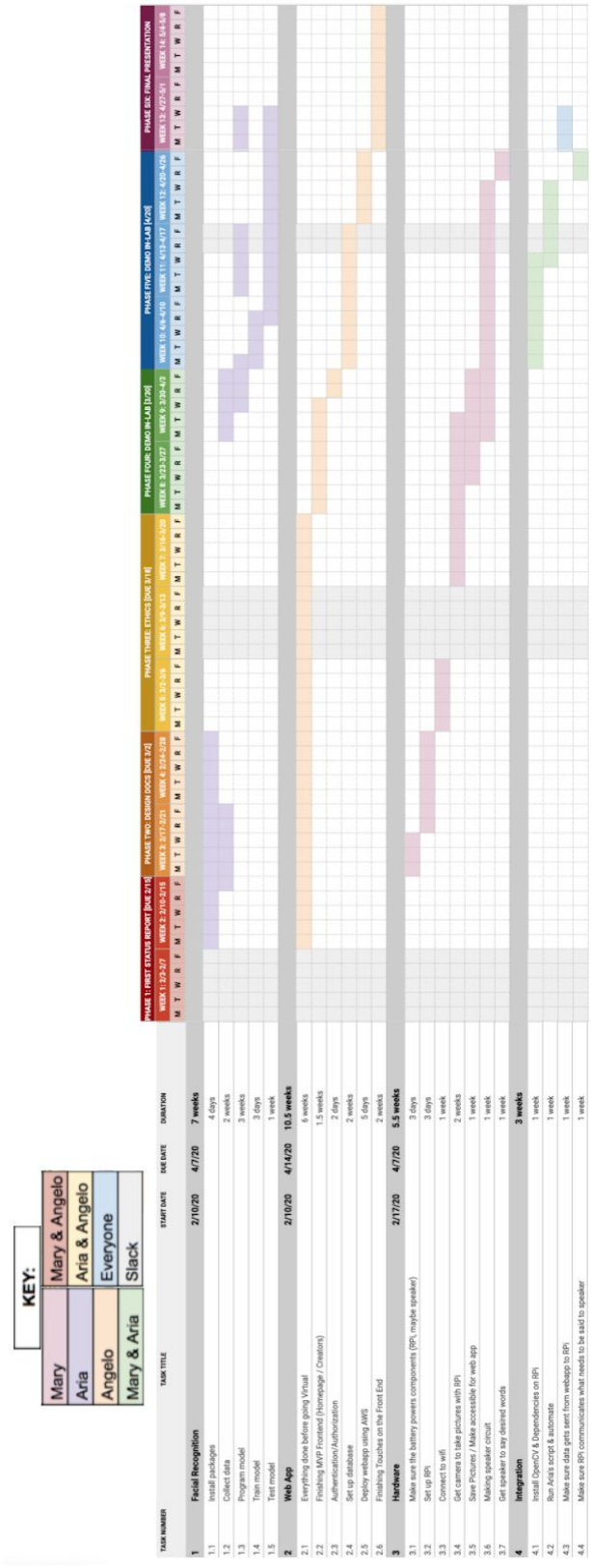


Fig. 5. Gantt Chart Schedule and Key

C. Budget

Parts	***Quantity***	***Used***	***Price***
Raspberry Pi 4 Model B 2019 (4GB)	2	✔ ☒	$61.95 ea
32GB Raspberry Pi Preloaded (Raspbian) SD Card	2	✔	$19.99 ea
Arducam ¼ Inch 5 Megapixel Sensor Spy Camera Module for Raspberry Pi	1	✔	$26.99 ea
Raspberry Pi Camera Module V2-8 Megapixel, 1080p	1	☒	$22.99 ea
Black Flex Cable for Raspberry Pi Camera - 2 ft	1	✔	$5.80 ea
CSI to DSI Cable Extender	1	✔	$2.95 ea
Micro HDMI to HDMI cable Adapter	2	✔ ☒	$7.98 ea
USB Type C to USB 3.0 Cable - 3 ft	1	✔	$ 7.99 ea
3.5mm Male Plug to Bare Wire Open End TRS 3 Pole Stereo	1	☒	$7.80 ea
CanaKit Raspberry Pi 4 Power Supply (USB C)	1	✔	$9.99 ea
Bone Conductor Transducer with Wires - 8 Ohm 1 Watt	1	☒	$8.95 ea
Stereo 2.1W Class D Audio Amplifier - TPA2012	1	☒	$9.95 ea
USB Battery Pack for Raspberry Pi - 10000mAh - 2 x 5V outputs	1	✔	$39.95 ea
Ethernet Cables	1	☒	$6.67 ea
SanDisk Extreme 32GB MicroSDHC UHS-3 Card	1	☒	$10.29 ea
Shipping			$16.44
Total			$340.16
Budget			$600

D. Risk Management

Our biggest risks and unknowns came from the facial recognition in our project. We believed our accuracy may be a little low because we are testing on a similar data pool, with our main data coming from the three of us and the main cast of Jurassic Park. Due to this, we believe that we might have a skewed accuracy, false positive, and false negative statistics. We are trying to mitigate this by having a larger and more diverse pool of data with people of all genders, ethnicities, and ages.

Another risk we are facing is the user experience of the device, and the fact that our main users are going to be older, so making a web application that is accessible and intuitive to them is a challenge.

VIII. RELATED WORK

The project most similar to ours thus far is the Raspberry Pi facial recognition project in our base API. Similarly to our project, theirs uses a Raspberry Pi 4, the Haar Cascade algorithm, and OpenCV to identify and verify individuals in their database. In addition, the recognition process in the API is identical to ours. The recognition process we are using begins by capturing images and framing the faces in them. The faces are detected using the Haar Cascade algorithm and the face embeddings are created. The face embeddings are then compared to those in the database and the faces are recognized if they exist, or recognized as Unknown if they do not exist. The most unique part of our project, in comparison to the base API, is the addition of the speaker and our overall goal to ultimately ease the lives of those diagnosed with dementia.

IX. SUMMARY

Our system did in fact meet our design specifications, with the exception of the speaker portion of our project and the timing with which interactors are recognized. If we had more time to improve the system performance, we would have liked to improve the hardware, to maximize user comfort.

A. Future work

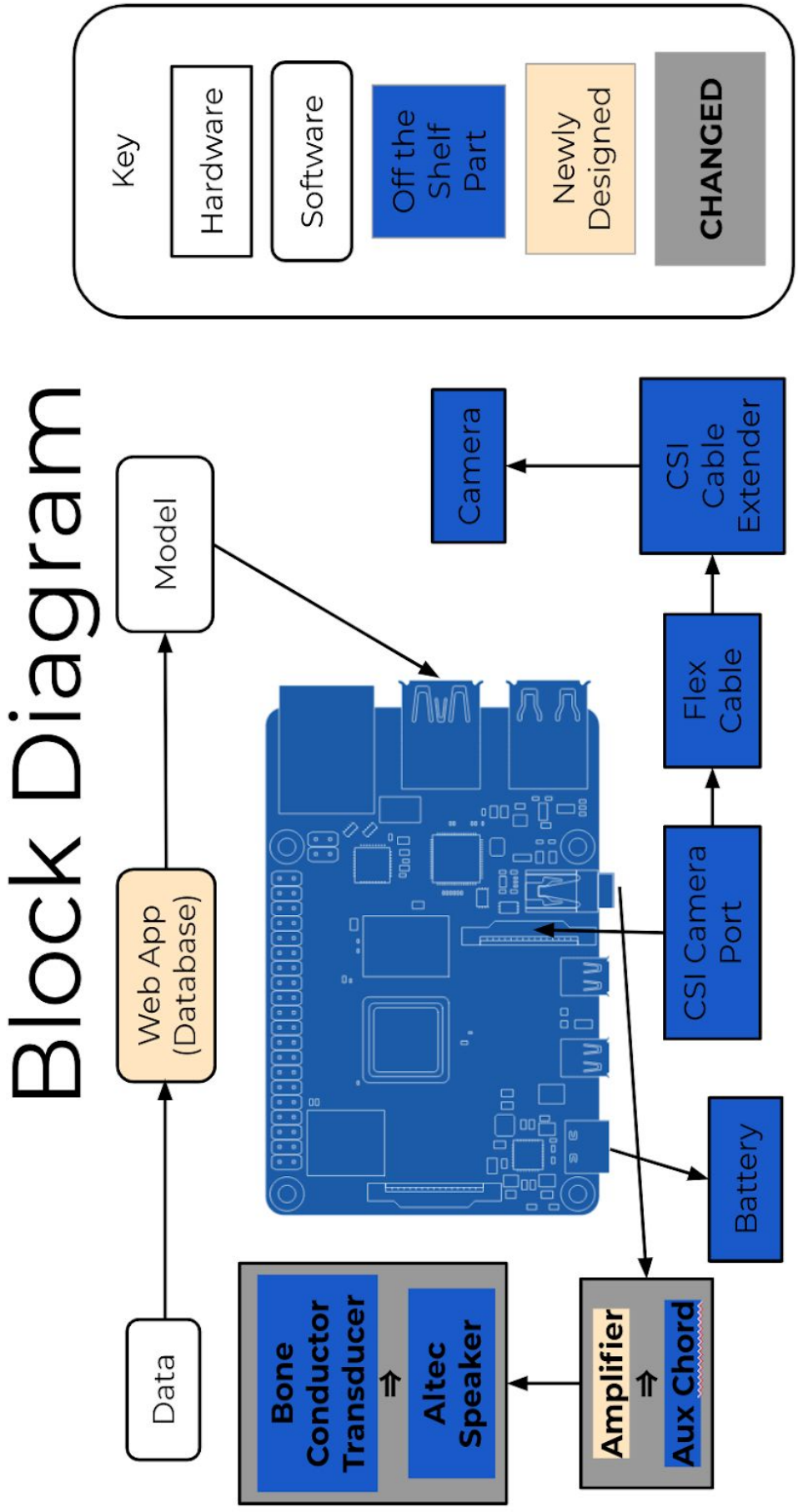
We do not plan to work on our project beyond the semester at this point in time. However, if this plan changes, we would most likely change the hardware to improve user comfort by minimizing the wires and connections to the Raspberry Pi. In addition, we can see if we can improve the speed at which interactors are recognized.

B. Lessons Learned

Throughout this past semester, we have learned many lessons. Our greatest challenge was the transition to virtual learning and thus no longer having in-person time to work with our group members. However, virtual meetings were much easier than anticipated and went quite smoothly, with the exception of ease of debugging. We would recommend to other student groups that may want to address this application in future semesters to plan as much ahead as possible, and leave lead time for parts to arrive, since this took much longer than we planned for. If a group is planning to complete a similar project, we recommend researching installation guides and tutorials during beginning stages of the project.

REFERENCES

1. https://www.frontiersin.org/articles/10.3389/fpsyg.2012.00454/full
2. https://docs.aws.amazon.com/polly/latest/dg/limits.html
3. https://www.amazon.com/Arducam-Megapixels-Sensor-Camera-Raspberry/dp/B012QRGUCQ/ref=cm_cr_arp_d_product_top?ie=UTF8
4. https://www.amazon.com/gp/product/B07J68TJ7L/ref=ox_sc_act_title_1?smid=A10G48GIGTLVM0&psc=1
5. https://www.adafruit.com/product/3671?gclid=Cj0KCQiAnL7yBRD3ARIsAJp_oLZWwgNqoUQcNIraA6gVWo-clBcQVDEBM-vuAFRkb4MbO1e-acGHzV8aAjRBEALw_wcB
6. https://www.amazon.com/Raspberry-Model-2019-Quad-Bluetooth/dp/B07TC2BK1X/ref=asc_df_B07TD43PDZ/?tag=&linkCode=df0&hvadid=380013417597&hvpos=1o1&hvnetw=g&hvrand=2842360819116105 99&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=9005925&hvtargid=pla-781430589105&ref=&adgrpid=77922879259&th=1
7. https://www.adafruit.com/product/1674?gclid=EAIaIQobChMIloSO0aan5wIVFKSzCh1LggaiEAQYAiABEgJwk_D_BwE
8. https://www.amazon.com/Fancasee-Replacement-Connector-Headphone-Earphone/dp/B07Y8KGJKL
9. https://www.adafruit.com/product/1552
10. https://www.adafruit.com/product/1566


Block Diagram
Key
Hardware
Software
Off the Shelf Part
Newly Designed
CHANGED
Model
Web App (Database)
Data
Camera
CSI Cable Extender
Flex Cable
CSI Camera Port
Battery
Bone Conductor Transducer
Altec Speaker
Amplifier
Aux Chord