Facial Recognition Lockbox

Ryan Wiegman, Julian Boaz, Che' Baptiste, Bryce Dere

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

***Abstract* – The objective of this project is to create a facial recognition lockbox. With online shopping growing at an ever-steady rate. Our main goal in designing this product is to use the advanced technology of our time to decrease the amount of porch thefts at-home shoppers experience. Using a highly accurate artificial intelligence program, that is pre-trained through Amazon Web Services, a camera will scan the face of the user either allowing access to the box or not. Users will be connecting to the box seamlessly through a mobile application for setup and to allow pictures to be uploaded for comparison to our database. With the Lockbox in current development we are very hopeful in the security, technology, and cost efficiency of our product.**

Index Terms – artificial intelligence, image processing, object analysis, user interface

I. INTRODUCTION

With technology increasing at an incredible rate and the want to make products more accessible to everyone, online shopping is at an all time high. However, with the increase in technology the products to protect the items bought online have not improved all that much. This is where we intend to design a highly technological and cost efficient lockbox.

As mentioned earlier, online shopping is increasing at a drastic rate with companies like amazon and even food companies like Uber Eats and DoorDash. With an estimated $870.8 billion dollars spent in e-commerce sales and with more than 40% of Americans having been victims of package theft. The amount of porch pirates are increasing as quickly as online shopping is. Therefore, the need to protect your purchase is becoming more and more important. However, with the current lockboxes on the market, if you are looking for one implementing smart technology, it could cost you hundreds and even thousands of dollars. Then there are Lockboxes that will not cost you as much but provide less security, some only including a simple padlock loophole.

Our solution to this is to design a lockbox that implements facial recognition and an artificial intelligence program to unlock potential users. Facial recognition is not all that new anymore, most commonly used in cell phones to unlock, but there is no lockbox on the market that has implemented it. Using this we hope to accomplish a way to make a facial recognition lockbox that is not only very secure but also seamless for the user to operate.

II. EXISTING SOLUTIONS

In attempting to design a solution to package theft and maintenance outside of a home, it is worth examining solutions that have already been implemented and to what degree they were successful.

The official solution on the part of most courier organizations to combat the problem of package theft is to require a person to accept the package at the point of delivery rather than leaving it unattended outside the building, and to collect a signature to verify that this was done.

Requiring a signature from someone handed the package is usually only done for deliveries where the value of the package exceeds a certain amount, and though this can be effective, it poses a variety of logistical complications and can be very impractical if the volume of delivered packages exceeds an occasional delivery.

Perhaps the most common and most straightforward solution to package theft that does not involve direct supervision of the delivery to a person is to attempt to simply hide the package itself in the vicinity of the door, such as under a mat or behind a bush, to attempt to conceal the package itself from anyone who may wish to take it if they knew of its presence.

However, with a closer examination, we find that this is a very ineffective solution. Many packages are

simply too large to be adequately concealed from anyone looking even from a short distance away. Furthermore, if there are no simple options with which to conceal the package, such as in an apartment building, or in an easily visible location, concealing the package is not as straightforward as it could be otherwise, and it will certainly not be as effective.

Therefore, we see that this straightforward solution is not really a solution at all, and we will require more invested design solutions to address this problem. With that in mind, let us examine some of these more invested existing solutions.

Existing Package Lockbox Implementations

Traditional lock-and-key boxes would be the next logical step when considering options to secure parcel deliveries, and the ability to initially implement such a solution is very high, as many such products exist and have done for some time.

However, attempting to actually make use of this solution becomes more complicated the more versatile it is expected to be; arranging for a particular courier service to have access to a copy of a key may be feasible for some commercial applications involving delivery of small volumes of items, but when the prospect of providing such keys to multiple courier services or attempting to scale this application to residential uses is considered, this solution rapidly becomes very impractical.

The next logical option for a potential solution might be a simple electronic lockbox that utilizes some type of network connectivity to have a rotating credential used to unlock itself. This solution would have the best versatility of any that have been considered so far, and indeed, this is the one that our solution primarily attempts to improve upon.

Eufy Security SmartDrop (left) & Yale Smart Delivery Box (right)

Seen above are two such products that implement this solution: they both use internet connectivity to authorize credentials through a smartphone app and/or a Personal Identification Number entered through an on-device keypad.

These products are both powered by onboard rechargeable batteries, and cost between three and four hundred dollars. These can both be considered limiting factors for the effectiveness of the implementation of this solution, as the requirement for regular maintenance of the power for the box runs counter to the goal of making the use of the box more convenient for the user and the higher price point limits the scalability of the solution, especially for residential uses, for which minimizing the price point is a high priority design requirement.

How is Our Solution Different?

The primary goals of our solution to courier delivery security are: 1) to make the lockbox as secure as possible while also being as easily accessible for authorized users as possible, 2) to minimize the cost to the user of the box, and 3) to minimize unnecessary friction during use (routine maintenance, time spent setting up app or configuring box during each use, etc.).

We aim to achieve these goals by building a lockbox utilizing a facial recognition algorithm connected to the internet via a microprocessor to authorize the unlocking of the box with little to no input from the user. The box will have an onboard microcontroller in addition to the microprocessor to control peripheral elements.

Furthermore, to make the box more accessible to more use cases, we have implemented a heating element into the box to be used in cases of food delivery to keep food warm and safe to consume.

What follows is an explanation of the implementation of these methods and goals into this solution.

III. REQUIREMENT SPECIFICATIONS

For the project to be successful, we included some requirement specifications to guide our project. The requirement specifications include:

Cost: The goal of our group is to create a project that meets the required engineering specifications of the class without exceeding more than $300. The reason for this is because our group does not have any

sponsors, meaning this entire project is independently funded by 4 college students. As such, we do not have much money to spare, so our choice of electronic components were selected with that in mind.

Activation Method (Drop Off): This is just one of the 2 methods available for using the Facial Recognition Lockbox. This method will not require the use of any kind of interface. An individual needs to be able to walk up to the Lockbox and be able to open and close it when there is not an object residing in it. When an object is detected inside of the lockbox, it will need to close and stay closed until the user opens it.

Activation Method (Pick Up): This method requires a series of steps in order to go through the security features of the lockbox. After a package has been dropped off, the owner will be able to pick it up by using the mobile application on their phone and by physically interacting with the Lockbox. By uploading a picture of themself through the mobile app and using the 6 digit code also received through the app, the user will be able to unlock the box.

Activation Range: Due to the fact that this is a Facial Recognition Lockbox, you will have to be in very close proximity to the Lockbox to use all of its functions. The owner of the Lockbox will be able to upload a picture anywhere from the mobile app, however, in order to pick up the package you will have to be next to the Lockbox.

Activation Speed: The Facial Recognition Lockbox is a very software heavy project, as such we hope to make the time for the entire process as fast as possible. There are a few aspects of such software that we aim to achieve. The time between clicking a button on the app and it performing any kind of api call should be almost instant. For example, when making a new account it should instantly show up in the database. We aim to have the Facial Recognition Software to work within 5 seconds of it starting, while the whole process from the user input on the app to the Raspberry Pi should be 7 seconds.

Power Consumption: The raspberry pi needs about 5 Volts to function, while the MSP430 needs about 3.3 Volts to function. In addition to the other electrical components of the Facial Recognition Lockbox, our aim is to have the consumption of our project be less than 10 Volts. The power needed to supply both of the boards come from the power supplies specifically made for their respective system.

Facial Recognition Accuracy: As this is the main way that our project will differentiate whether or not a person is allowed to access the Facial Recognition Lockbox or not, our facial recognition software has to be very accurate. As a result, we will consider the accuracy of our software adequate if it is able to correctly identify the same individual between 2 images at least 80% of the time.

REQUIREMENT SPECIFICATIONS

Engineering Specifications	Marketing Specifications
Cost < $300	Easily Recreatable
Power Consumption < 10V	Low Cost
Activation Speed < 5s	High Reliability
Accuracy > 80%	High Reliability

Above is a summary of all the specifications previously discussed as well as the marketing reasons for these design decisions. Our overall goal for these specifications is not only to make an inexpensive and effective project, but also to demonstrate the knowledge that we have gained in our studies thus far, as well as our ability to work as a team in order to bring a concept to reality.

IV. SOFTWARE IMPLEMENTATION

When mapping out the software of our product, we wanted to ensure it was as lightweight as possible. Keeping this in mind, we found that Amazon Web Services provided many tools that would allow us to achieve this. We laid out the type of processes that would be used in our product's workflow which are facial recognition, and storage of user accounts with profile pictures through a mobile application. AWS provides four services to easily implement these processes called AWS Rekognition, AWS Cognito, AWS Amplify, and AWS S3.

With this product being very software heavy, in artificial intelligence and having a mobile application we needed the proper framework to get it started; the services from AWS provided just that. Rekognition is

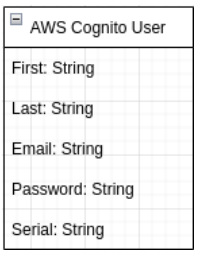
one of AWS services that focuses on machine learning and computer vision. Allowing us to customize computer vision API calls that can be directly implemented into our application or even use pre-trained modals, all centered around using artificial intelligence to analyze images. Another benefit Rekognition provides is its ability to scale up or down with the needs of the program.

AWS cognito is the next framework that our group will be implementing in our application. Cognitos main focus is on access control for websites and mobile applications, which makes it perfect for our project in terms of security and functionality. Dealing with user sign-up and sign-in, cognito gives us the type of security on the application side that we need to protect sensitive information given by the users. Along with AWS Rekognition, Cognito also is scalable to the needs of the program, being able to handle millions of user accounts.

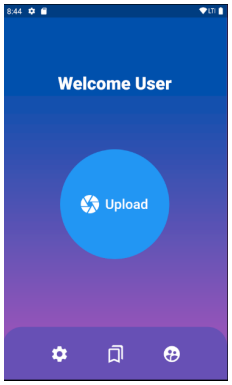
Connecting alongside Cognito, AWS Amplify will be used inside the application as well. AWS Amplify's main use is for building front-end and backend parts of websites or mobile applications. However, it also has authentication properties that make it perfect for combining with Cognito and the objective of our project.

The last service we will be using is AWS S3 buckets. This is where all of our user pictures will be stored, making this one of the most important parts of our design. The main benefit of using a S3 bucket is its ability to scale with a project and still hold all of its data durability. S3 buckets also hold a level of security to them, allowing us to protect user data as best as possible. Finally, S3 allows up to 5 GB of free data; and with the Facial Recognition Lockbox still in the prototype stage, it allows us to save on unnecessary costs.

The software workflow starts with the mobile application. The mobile application is written in Dart which was Che's language of choice for writing apps given prior experience. In order to link the front end of the application to our AWS account, AWS permissions were set up under an IAM profile which only gave explicit permission to access AWS Cognito, AWS S3, and AWS Amplify. In order to maintain security, this IAM profile does not have access to AWS Rekognition. The IAM profile has an access key and secret access key which is required to make API calls to AWS. AWS Amplify allows us to configure the app to make API calls through that IAM profile. Once the user downloads the application, the user will be prompted with a sign up screen which has a few different text fields. Their first and last name, their email, the serial number of their lock box, and a password of their choice.



Once they've clicked the sign up button (assuming their passwords match), they will be prompted with a verification screen in which a 6 digit code will be sent to their email. When they verify their email they can login to their account, and upload a profile picture. They can also change their profile picture at their leisure in case they go through a physical change or they would like someone else to have access to their lockbox.



Mobile Application Home Screen

As it can be seen above, the mobile application provides the users with an easy to navigate home page. As previously mentioned, the big upload button in the middle will prompt the user to upload a picture of whoever is going to access the Lockbox. Included in the mobile application is also a settings page and an about us page for the creators of the Facial Recognition Lockbox.

Once an item has been placed in the lockbox and the door closes and locks (this will be explained further in the hardware section of the document), there will be a Python script that is executed on the Raspberry Pi. Similarly to the mobile application, there is an IAM profile that was created for the microprocessor specifically that has permissions that allows it to access AWS Rekognition, AWS Cognito, and AWS S3. The Raspberry Pi can be configured with this IAM profile via the AWS CLI version 1. Unfortunately given that the Raspberry Pi's operating system is 32bit, AWS CLI version 2 cannot be utilized. Even though version 2 of the AWS CLI doesn't provide more functionality for our purposes, version 1 isn't officially supported any longer and so in the future if we require more functionality, we will have to upgrade hardware to achieve this. Given that the AWS CLI is configured to the IAM profile set up for it, the Python script uses a library called Boto3 in order to make API calls.

Before making API calls to AWS, the Python script will utilize a library called OpenCv in order to take a photo of the user. This photo will be taken once the user makes a keystroke on the keypad as the first part of the script.

```
ret, frame = video.read()
cv.imshow("photo", frame)
k = cv.waitKey(1)
if (k%256 == 27):
    print("Exiting...")
    break
elif k%256 == 32:
    print("Photo taken")
    cv.imwrite("sample.png", frame)
```

Once this photo is taken, the first API call to AWS Rekognition will be made. The method "detect_faces" will be called on the photo, which will determine how many faces are in the photo.

```
    response = client.detect_faces(Image={'Bytes': imageTarget.read()},
Attributes=['ALL'])
```

If there are more or less faces than exactly one, it will wait for the user to input another keystroke and take another photo. If the photo contains exactly one face, then the script will make an AWS Cognito API call called "admin_get_user" which will grab the email associated with the serial number that is attached to the box, which will be used to locate the profile picture and retrieve that from the S3 bucket.

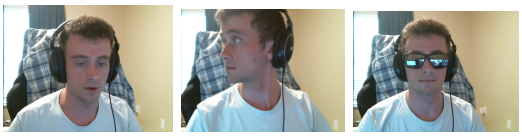
```
    response = client.admin_get_user(UserPoolId=userpoolid,
Username = username)
```

Finally, the script will make an API call to AWS Rekognition with the method "compare_faces" which will compare the photo taken locally and the photo downloaded from the S3 bucket.

```
    response = client.compare_faces(SimilarityThreshold=99,
SourceImage={'Bytes': imageSource.read()}, TargetImage={'Byt
es': imageTarget.read()})
```

If the facial recognition model is 99 percent certain that the two faces match, it will send a signal to the microcontroller, otherwise, it will return to the beginning of the script in which it waits for another photo to be taken by the user.

The AWS Rekognition facial recognition model is incredibly accurate given any style of photo. We've tested the model on photos of different lighting conditions, different angles, sunglasses, different backgrounds, and haircuts. All of these matched over the 99 similarity threshold that we set as a requirement. Below are three of those examples:



Once the box has been unlocked via confirmation of the user's face, the overall process of the software is complete. The door will remain open for the next delivery to take place, and again, when the door is shut, the process is started over.

It's important to note that users are not restricted to only have a picture of themselves associated with their account. They can choose to upload, for example, a family member's or friend's face if they are out of town, and would like someone else to pick up their delivery. A user could have their child's face on the account in case they would like to order food for their child to receive if they are home without their parents. The user's photo can only be changed once they have logged in within the mobile application. Given this fact as well as AWS being the only Department of Defense approved hosting service, we are comfortable with the security provided to the user.

Fortunately, given that we are using AWS Rekognition for our facial recognition process, the hardware doesn't bottleneck the speed at which a user's identity is confirmed. This does mean that a requirement for utilizing our product is having a

consistent internet connection to make API calls. Internet speed will not hinder how fast the model makes predictions, however, if the connection is interrupted, no prediction will be made at all.

V. HARDWARE IMPLEMENTATION

Raspberry Pi Microprocessor

The Raspberry Pi was the chosen microcontroller unit (MCU) for our project. There are a variety of reasons that we decided upon the Raspberry Pi for this project, but the main reason was due to the cost. One of the members of the group happened to have a spare Raspberry Pi that they were not using. As a result, we would be able to utilize the Raspberry Pi for our Facial Recognition Lockbox for no cost, allowing us to distribute what would have been a minimum of over 100$, to other resources. The Raspberry Pi is a bit of overkill for the scopes of this project, however, it does everything we need the MCU of our project to do while also being the cheapest option available for us.

Quad Core Processor: The Raspberry Pi 4 comes equipped with a 64-bit quad core processor. This enables the Raspberry Pi to do a large variety of things, including handling I/O ports, an integrated wifi module, an camera module that connects directly to the board itself, and the ability to display whatever you may want it to display all at the same time. Additionally, the Operating System specifically made for the Raspberry Pi (Raspbian) allows us to call different python scripts when needed, making it possible to run Amazon Web Service API calls remotely. All of these features are necessary for our project to succeed and with the quad core processor that the Raspberry Pi possesses, it is able to handle all of those tasks and then some.

Integrated Wi-Fi Module: The Raspberry Pi comes equipped with a 5 GHz Wi-Fi module installed on the board itself allowing it to access local networks. This function is essential for our project, as the Raspberry Pi needs to be able to communicate with Amazon Web Services in order to not only grab the image from the mobile application, but to also communicate with the facial recognition software that lives there to let the Pi know whether or not to open the box for the person standing in front of it.

Input/Output Ports: The Raspberry Pi comes equipped with multiple USB ports. This allows us to connect multiple things, including a small screen and keypad for the user to interact with when they walk up to the Facial Recognition Lockbox.

Camera Module: The Raspberry Pi also has the ability to connect a certain type of camera directly into the board. This camera not only uses less power but also takes higher quality pictures compared to a USB camera due to the fact that the USB camera cannot connect to the CSI Interface like the camera module, meaning it takes more work for worse quality with the USB camera. The camera module we purchased for this project can take a still picture with a resolution of 2592 x 1944.

Picture Taken By Camera Module

As you can see, the camera module is able to take a clear picture of whoever is standing in front of it. In addition to this picture, multiple other pictures were taken in a variety of different lightings. Given that the individual's face is visible, the facial recognition software was still able to meet our threshold of an 80% accuracy.

Microcontroller Unit (MCU)

In addition to the Raspberry Pi, the device will have an onboard microcontroller to trigger the connected peripherals. This includes the solenoid locking mechanism and the heating element, as well as sending trigger signals to the Raspberry Pi microprocessor when any buttons are pressed.

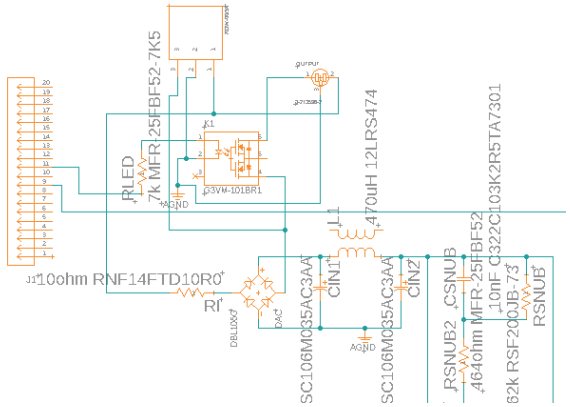
The microcontroller used in our implementation is the MSP430FR6989. The MSP430 microcontroller family is a very commonly used group of microcontrollers that contains a range of different versions. The FR6989 type comes equipped with 2KB of memory, 16MHz of processor speed, 83-pins, and an output voltage of between 1.8V - 3.8V. The MSP430FR6989, however, does not come with a USB connector, which means we are planning on

connecting the controller to the Raspberry Pi microprocessor using its GPIO pins. The board also is relatively cheap, costing around $30 USD.

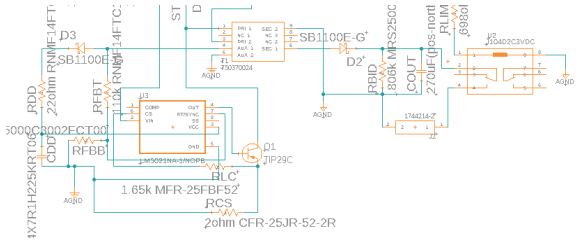
However, one of the biggest reasons for our group in choosing this particular controller is that one of our members already owns one. This allows us to save on time and production cost, giving us the opportunity to invest that money elsewhere. Finally, we all have experience working with the MSP430FR6989 from our previous embedded system class, making it ideal as a high level controller for other electronic components.

Printed Circuit Board & Peripherals

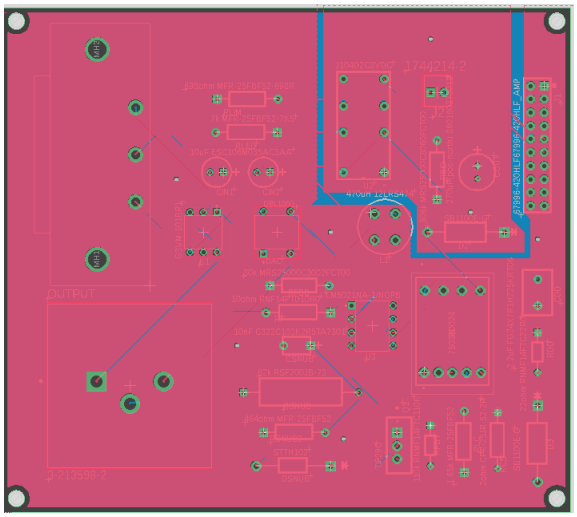
In between the microcontroller and the peripherals is a printed circuit board that contains power input and output for connected components, pin connections to interface the those components with the microcontroller, as well as a power conversion circuit intended to rectify incoming Alternating Current power from the connected wall outlet into Direct Current power at the correct voltage and current for the solenoid to allow it to be used.



Seen above is part of the schematic for the printed circuit board, including part of the power conversion components (which are in a flyback topology), as well as the AC input and output, the relay for the power for the heating element, and the pin connections from the microcontroller.



The rest of the circuit contains the controller unit for the power conversion, the transformer to convert the voltage for the solenoid, and the switch and pin output for the solenoid connection. Of interest in this part of the schematic is the isolated ground on either side of the transformer, which provides some buffer between the two voltage regions. This concept was implemented on the board design by isolating the ground pour for each region on the top side of the board before connecting them to the pour on the other side with vias, as can be seen in the top right of the image below.



The solenoid itself is attached directly to the latch on the lockbox and is locked when unpowered, meaning the lock fails closed. This is the ideal position for our hardware requirements, as the lock will spend the majority of its time being locked when in use, and will stay that way if someone attempting to breach the box unplugs it.

Because the locking mechanism is a solenoid specifically, this means that there is non-negligible heat generation when it is unlocked (the solenoid in our implementation is a 12V, 0.54A component, meaning it has a power dissipation of ~6W). Therefore, the design for the microcontroller will have specific safeguards to ensure the box doesn't remain unlocked for too long and ensure the longevity of the components.

As mentioned briefly, this solution has an additional goal of being applicable specifically to food deliveries. We feel that this application aligns well with the design aspects of our solution, as food deliveries tend to be somewhat more frequent among existing customers, as well as taking place over a

much shorter timetable than a traditional package delivery.

To facilitate the delivery of food, our solution aims to include a resistive heating element in the interior of the lockbox to keep any hot prepared food left inside at a safe temperature that is optimal for human consumption. This resistive heating element is also controlled by the microcontroller via the relay component on the printed circuit board, and will only turn it on when it receives the relevant signal from the microprocessor.

Finally, all of these components are configured to attach to a physical lockbox, which will be large enough to hold mid-size package deliveries with enough room left over to hold the required electronics and the heating element. Our implementation uses a pre-existing lockbox purchased online that comes with a latch to which the solenoid is attached, and then any fitting or cutting can be performed to accommodate the microprocessor and controller, the printed circuit board, the heating element, and routes for power connections.

VI. CONCLUSION

As stated porch pirates are becoming more and more of an issue with people everywhere experiencing having their packages stolen. We believe that the product that we are building is a great way to combat this issue. Using our facial recognition AI program, our mobile application, database and state of the art hardware, we are providing security, technology, and an easy way to install and use.

VII. BIOGRAPHY

Ryan Wiegman is currently a senior at University of Central Florida, working towards completing a degree in computer engineering. Ryan's future goals after graduating is to find a job in software development, with a focus on artificial intelligence.

Julian Boaz is a 23 year old Computer Engineering student attending the University of Central Florida. Julian prefers working on software and specifically machine learning and artificial intelligence. He has accepted a position at Lockheed Martin working on the training simulator for the F-35 project at Rotary and Mission Systems in which upon graduating he will begin full time.

Bryce Dere is a senior studying Electrical Engineering on the Signal Analysis & Communications Track at the University of Central Florida. He is graduating in the Summer 2022 semester and is planning to seek employment in the development of music technology.

Che' Baptiste is a computer engineer hoping to continue polishing his programming skills while working with LexisNexis. He plans on continuing his education by earning a master's degree in computer science soon after graduation.

VIII. REFERENCES

[1] https://www.emarketer.com/content/us-ecommerce-forecast-2021

[2] https://www.crresearch.com/blog/2020-package-theft-statistics-report

[3] Schütz Julia. (2011). Kontinuierliche versus Diskrete Modelle der rekognition und des quellengedächtnisses. Amazon. Retrieved July 15, 2022, from https://aws.amazon.com/rekognition/

[4] Roose, H. (1987). Cognito. Amazon. Retrieved July 15, 2022, from https://aws.amazon.com/cognito/

[5] Hollands, M. (2015). Amplify. Amazon. Retrieved July 15, 2022, from https://aws.amazon.com/amplify

[6] Strand Street Press. (2002). S3. Amazon. Retrieved July 15, 2022, from https://aws.amazon.com/s3/