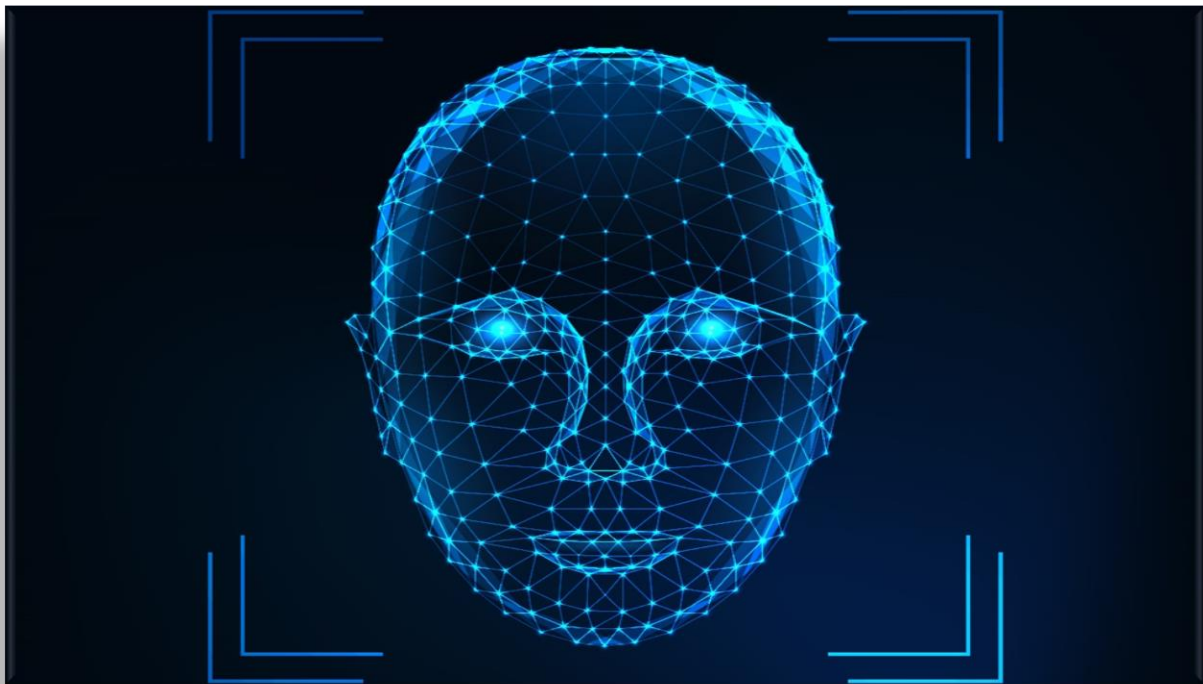


Middlesex University

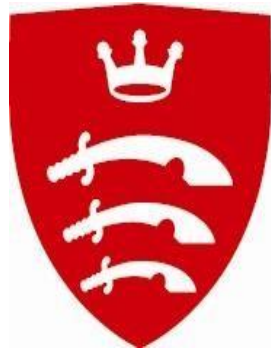
School of Science and Technology

CST3990 **Software Development Project**

Autumn/Winter term
2021/2022



Date:	30 April 2022
Supervisor:	David Gamez
Student Name:	Manjot Kaur
Student ID Number:	M00735429
Campus:	Hendon
Title:	Face Recognition using Raspberry Pi, zeroCam and Speaker



Middlesex University

School of Science and Technology

I hereby confirm that the work presented here in this report and in all other associated material is wholly my own work.

Signature:	Manjot Kaur
Student ID:	M00735429
Date:	30th April 2022

Acknowledgments

I would like to extend my deepest gratitude and appreciation to my supervisor, Dr. David Gamez, for his continuous guidance, mentorship, and tireless support regarding my project. As a result of meetings and conversations with my supervisor, I developed an objective and comprehensive assessment that enabled me to look at things from multiple perspectives.

Contents

Acknowledgments.....	3
Abstract	7
1. Introduction.....	8
2. Literature Review.....	8
2.1 History.....	8
2.2 Data Collection.....	9
2.3 Benchmarks	9
2.4 Libraries and SDKs, Web APIs.....	9
2.4.1 Open-source libraries and SDKs.....	10
2.4.2 Web APIs.....	10
2.5 Facial Recognition Application areas.....	11
2.5.1 Prevent retail crimes	11
2.5.2 Unlock Phones.....	12
2.5.3 Smarter advertising.....	12
2.5.4 Find missing person.....	12
2.5.5 Law enforcement	13
2.5.6 Attendance System.....	14
2.5.7 Facilitate secure transactions.....	14
2.5.8 Social-media.....	14
2.6 Face Recognition Techniques	15
2.6.1 Face recognition based on Convolutional Neural Networks	15
2.6.2 Holistic Matching.....	16
2.6.3 List of techniques.....	16
2.7 Factors affecting accuracy of Face Recognition	17
2.7.1 Ageing	17
2.7.2 Facial expression.....	17
2.7.3 Plastic Surgery.....	17
2.7.4 Occlusion.....	17
2.7.5 Low Resolution	17
2.7.6 Noise	17
2.7.7 Illumination	17
2.7.8 Pose variation.....	17
2.8 Limitations of current face recognition technology	18
2.8.1 Poor Image Quality	18
2.8.2 Small Image Sizes	18
2.8.3 Data Processing and Storage	18
3. Application Requirements and Design Specifications	19
3.1 System Description	19
3.2 Stakeholders	19

3.3 Functional Requirements.....	19
3.3.1 The User's perspective	19
3.3.2 The System's perspective	19
3.3 Use Case Diagram.....	20
3.4 Activity Diagram	20
3.6 Data Flow Diagram-	21
3.7 Database Design.....	22
3.8 Hardware	23
3.8.1 Raspberry pi with laptop	23
3.8.2 Camera attached to Raspberry pi	24
3.8.3 Uploading Pictures from the Raspberry pi to AWS S3 Bucket.....	24
3.8.4 Green LED connected to Raspberry Pi	25
3.8.5 USB to Audio connected to Speaker for Audio Output	25
3.8.6 Text to Speech.....	25
3.8.7 Raspberry pi and camera mounted on sunglasses	25
3.9 Wireframes.....	25
3.9.1 WebPage 1(View all the recognised Images).....	25
3.9.2 Webpage 2 (View new recognised Image).....	26
3.10 API Design	27
3.10.1 WebSocket API.....	27
3.10.2 Rest API	27
4. Implementation.....	28
4.1 Overview	28
4.2 Final Product.....	28
4.3 API Gateway	28
4.3.1 WebSocket API.....	28
4.3.2 REST APIs	29
4.4 System Architecture	29
4.5 Hardware Description	31
4.6 Application Description	32
4.7 Amazon Rekognition	32
5.Evaluation and Testing	34
5.1 Extensive review	34
5.2 UI Testing.....	34
5.2.1 HTML Validation	34
5.2.2 CSS Validation.....	34
5.3 Unit and API Testing	35
5.3.1 Lambda Functions Evaluation.....	35
5.3.2 API Gateway testing Interface.....	35
5.3.3 Rest API evaluate with Postman	36

5.3.4 Mocha and Chai Testing	37
5.4 Evaluation against requirements	37
5.4.1 Hardware Evaluation.....	37
5.4.1 Web Application Evaluation	38
5.5 Text-to-Speech Testing	38
5.6 Systematic Testing	38
5.6.1 Evaluation with Single Face Image clicked by zeroCam	38
5.6.2 Evaluation Using Photographs of Celebrities in the Crowd.....	39
5.7 Field Testing.....	40
6.Benefits	40
7.Conclusion.....	41
7.1 Ethical Reflections.....	41
7.2 Cloud Provider	41
7.3 Limitations	41
7.4 Future Work	41
7.5 Summary.....	42
8.References	43

Abstract

People are usually fascinated by celebrities. However, it is not always easy to spot a celebrity or someone who does not look the same at the time. It is possible that people are unaware of certain celebrities because they are not widely known, and thus they are unable to recognise them. Furthermore, looking everywhere at once is exceedingly tough and no one can see behind them. For example, to look for someone when there are people around, such as a family member or a friend. Similarly, a small percentage of the population is blind, making it difficult for them to see other people and objects clearly. However, it is not always simple to detect a celebrity when you are not actively looking. To address these issues, this project uses a zeroCam attached to a Raspberry Pi, as well as a speaker mounted on sunglasses, to create a facial recognition system. People can recognise celebrities in public or in crowded settings by utilising this equipment. The project utilises Amazon Cloud Services for face recognition and storing the dataset. After designing and implementing the hardware and web application for face recognition system, it was time to evaluate it, which was accomplished by using a variety of testing methodologies such as field testing (which proved that the hardware device worked and that the face recognition system ran properly), systematic testing (which shows the graphs of the system's accuracy), unit testing (which is done with Rest API with Mocha/Chai), and extensive testing to ensure the web application worked as intended. This project demonstrated that the hardware implemented for facial recognition system has the potential to be a valuable technology in the future.

1.Introduction

People often show a keen interest in celebrities. Nevertheless, it is not always convenient to detect a celebrity, or someone who might not look the same at that moment (i.e., a person without makeup, fake eyelashes, glasses, or hat/headband). It is possible that people are unaware of certain celebrities because they are not immensely popular, so they cannot recognise them. Additionally, it is extremely difficult to look everywhere at once. For instance, it is difficult to search for someone when there are a lot of people around, such as a family member or a friend. Similarly, a minority of people suffer from visual impairment, resulting in their inability to see other people and objects clearly. All these issues can be managed using a face recognition system. The use of face recognition is rising every day in several fields for example with the COVID-19 pandemic, facial recognition technologies are also increasingly used in digital health and disease outbreak prevention in combination with other forms of biometrics. (I-scoop.eu,2021). Considering these issues, this project has been implemented to help people recognise their favourite celebrities and loved ones in public. The project splits into two sections: hardware and web application. This project's hardware consists of a Raspberry Pi, a zeroCam, and Amazon Cloud Services. The Raspberry Pi powers the face recognition system. The zeroCam captures images and uploads them to the AWS S3 Bucket, which then runs face recognition using AWS Lambda and compares the uploaded image to the images in the database. The web application displays images of celebrity results generated by AWS image analysis. The raspberry pi is mounted on sunglasses, allowing people to wear these sunglasses in public and easily identify celebrities.

An overview of the literature is presented in the first section of this document. This chapter of the report provides a comprehensive analysis of previous research done on face recognition technologies. It covers different types of face recognition systems, the advantages and limitations of existing algorithms, and the core technologies of facial recognition. Which will also discuss the success and failure of this system. The third chapter describes the application requirements and design specifications, which include UML diagrams, database design, web application wireframes, and how this project is planned. The report's fourth chapter explains the Implementation, which describes the architecture of the entire hardware and web application, the final product, and the technology used to implement it. The fifth chapter describes the process of determining whether the implementation met all the requirements.

2.Literature Review

2.1 History

According to researchers, the average person can recognise about five thousand faces, though some super recognisers can memorise up to ten thousand. Facial recognition systems were primarily inspired by how humans recognise each other. In a face recognition system, digital images or videos are compared against a database of faces to make sure the faces match. The face recognition system created in the 1960s by Woodrow Wilson Bledsoe categorises faces by identifying their features (Wired,2020). The technology is more than fifty years old. When a research team ran experiments in 1964, they attempted to find out if computers could recognise faces through programming. To map the person's hairline, eyes, and nose, the team used an elementary scanner. The approach was not successful because computers have trouble recognising faces. However, the advancements in camera technology, mapping processes, machine learning, and processing speeds have led to an advancement in facial recognition. Using face recognition technology, Samsung was the first to utilise advanced facial recognition technologies in the Galaxy S4. This technology has improved tremendously over the past few years. Unlocking smartphones is the most common application of this technology. Since 2010, facial recognition technology has advanced rapidly, and September 12, 2017, was another significant breakthrough in the integration of facial recognition into our daily lives. This was the date on which Apple released the iPhone X, the first iPhone to support FaceID – Apple's marketing term for facial recognition. (nec.co.nz, 2020). Aside from this, Chinese companies use facial recognition technology to identify individuals at a crowd gathering. Face recognition has become

increasingly accurate after the application of deep learning approaches. DNA, fingerprints, voice pattern, and symbol recognition all fall under the biometric field of face recognition. Biometric technology such as facial recognition is less accurate than iris and fingerprint detection (bayometric.com,2020). Face recognition is being used for a variety of purposes, including crime prevention, event security, and making air travel more convenient. It contributes significantly to making the world a safer and smarter place.

2.2 Data Collection

Data collection for facial recognition allows users to analyse and compare the features of an individual's face. For effective training, thousands of better, millions of labelled images are required. Large Internet companies such as Facebook, Google, and Microsoft are not usually affected by this problem. Moreover, most of them offer free photo storage. Identification and authentication of a person are achieved through biometrics, which uses a set of data that is verifiable as well as recognisable and specific for that individual. The FaceNet collection at CASIA WebFace is the largest publicly available collection of labelled images, with about five hundred thousand images. Comparatively, the Google Facenet collection used two hundred million images. There was a significant improvement if the number of training images increased from thousands to a million. The Google FaceNet team published an analysis of the process of training their system. Furthermore, the authors suggest a solution to normalise the position and orientation of faces in images by using 3D rendering of images. Many face recognition systems use 3D rendering to do this.

2.3 Benchmarks

Among the most important benchmarks is the Labeled Faces in the Wild (LFW (Labeled Faces in the Wild)) dataset, which provides a dataset of faces from the internet. (paperswithcode.com). This dataset has been around for many years. As described in the standard LFW evaluation protocol, the verified accuracy is reported for six thousand face pairs in the LFW dataset. This dataset includes thirteen thousand two hundred thirty-three images of faces collected from the web. There are five thousand seven hundred forty-nine identities and one thousand six hundred eighty people with two or more images. On this dataset, several algorithms were evaluated. There was trivial difference in accuracy and results among the models on this dataset.

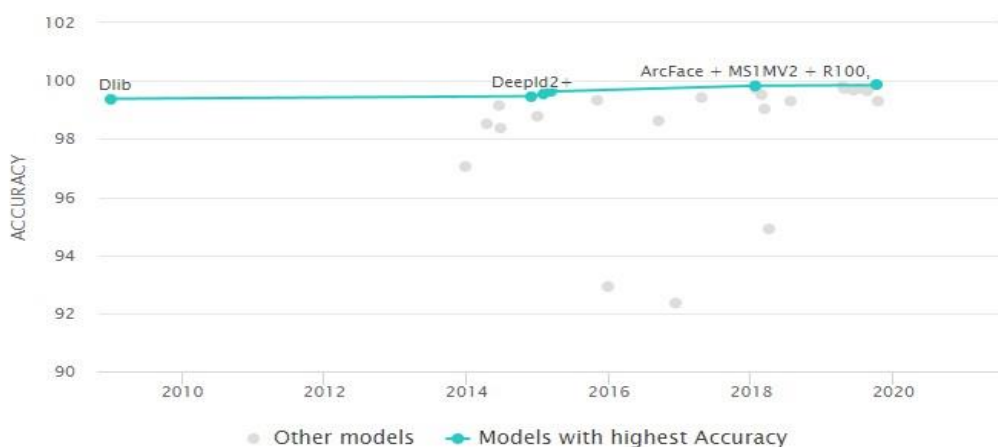


Figure 1 Labeled Faces in the wild (from LFW, 2020)

2.4 Libraries and SDKs, Web APIs

Presented here is a list of commercial face recognition APIs and open-source face recognition libraries/SDKs. While not comprehensive, it does give an overview of what is currently available.

2.4.1 Open-source libraries and SDKs

While many free and available libraries provide face detection, fewer can be used for face recognition as well. Below are some lists of libraries or SDKs that provide face recognition.

2.4.1.1 Openface

- The highest accuracy on Labeled faces in the wild (99.92%) which uses a face recognition library with mobile applications.
- The code is available on GitHub at: <https://github.com/cmusatyalab/openface>
- API Documentation: <https://openface-api.readthedocs.io/en/latest/index.html> - Programming language: Python

2.4.1.2 Libface

- No performance on Labeled faces in the Wild benchmark.
- The code is available at: <http://libface.sourceforge.net/file/Home.html>
- API Documentation: <http://libface.sourceforge.net/file/Documentation.html> - Programming language: C++

2.4.1.3 Cognitec

- No performance on Labeled faces in the Wild benchmark.
- The code is available at: <https://github.com/asterkin/phoenix/blob/master/cognitec-sdk>
- API Documentation: <https://www.cognitec.com/facevacs-technology.html>
- Programming language: C++, Java

2.4.2 Web APIs

2.4.2.1 Microsoft Computer Vision API

Using this Web API, the user will be able to analyse images and their features. 96 % accuracy is achieved with the high-level development algorithms for image processing. A free basic plan is available as well as several paid options, ranging from \$19 to \$199.

2.4.2.2 Lambda Labs API

In addition to detecting faces and identifying gender features, this API provides eye, nose, and mouth positioning information. It is highly accurate and allows for accurate and fast detection of faces. There are several paid plans available ranging from \$149 to \$1,449 per month, with Lambda Labs offering one thousand free requests per month with a free plan. The accuracy of lambda labs API is 99 % using LFW dataset. (quick-adviser.com, 2019).

2.4.2.3 Inferdo

The user can determine faces and genders, age, and facial features based on this machine learning algorithm. Inferdo tends to be nearly 99.97 % accurate in face recognition (recfaces.com,2021).

2.4.2.4 Face++

This algorithm is best for the extraction of facial information from images. Deep learning models are used to detect and analyse faces using the Face++ Face Detection API from Megvii, which is trained on millions of images to ensure maximum accuracy. A free version of Face++ is available. Paid packages range between \$500 and \$10,500, depending on the features. This API has 99 % accuracy.

2.4.2.5 Kairos

This API is best for finding faces and detecting features. In addition to defining gender and age, Kairos Face Recognition API also recognises faces in images, videos, and live streams. To prevent identity fraud and ensure secure communication with clients, Kairos offers an API application. Kairos is well known for its ethical approach to identity management. Its products support diverse communities around the world. There are several paid plans, starting at \$19 per month and going up to \$499 per month (recfaces.com,2021).

2.4.2.6 Amazon Rekognition

The Amazon Face Recognition API provides seamless integration with any of the platforms to detect and identify a person through images or live video recordings. Amazon Rekognition provides the ability to verify user identities by comparing their live image to a reference image. Amazon Rekognition detects Personal Protective Equipment (PPE), including face covers, head covers, and hand covers on people in images. Several Amazon Rekognition APIs are available such as-

CompareFaces- A face in the source input image is compared with each of the 100 largest faces detected in the target input image. If the source image contains multiple faces, the service determines the largest face and compares it with each face detected in the target image. This is a stateless API operation (amazon,2016).

DeleteFaces- An array of face IDs is supplied along with a collection ID to remove faces from the collection.

DetectFaces- This operation detects the top 100 faces in the image and returns face details, including the bounding box of a face, a confidence value (that the bounding box contains a face), and a set of fixed characteristics such as facial landmarks. The face-detection algorithm is most effective when applied to frontal faces. It is less effective for the detection of non-frontal or obscured faces (amazon,2016).

IndexFaces- Instead of saving actual detections of faces, Amazon Rekognition extracts facial features from the input image, and stores them in a backend database based on the underlying detection algorithm. The SearchFaces and SearchFacesByImage operations use feature vectors to match faces and search for people.

ListFaces- A face collection is represented by this listFaces. The function returns the metadata for faces in a specified collection. The metadata includes information such as bounding box coordinates, confidence, and ID.

RecognizeCelebrities- This function returns an array of celebrities recognised from the input image. The RecognizeCelebrities function returns the 64 largest faces on an image. It goes through the CelebrityFaces array to find recognised celebrities, and UnrecognizedFaces to find unrecognized celebrities. RecognizeCelebrities doesn't return celebrities whose faces aren't among the 64 largest faces (amazon,2016).

2.5 Facial Recognition Application areas

2.5.1 Prevent retail crimes

When known shoplifters, organised retail criminals or those with a history of fraud enter retail outlets, face recognition is utilised to swiftly identify them. Loss prevention and retail security experts can be promptly warned when a consumer enters a business that prevents a threat by matching their photographs against massive databases of offenders (FaceFirst, 2019).

2.5.2 Unlock Phones

Face recognition is currently used to unlock a range of phones, including the latest iPhone. This technology is a great technique to protect personal data and ensure that sensitive data is inaccessible to the perpetrator if a phone is stolen. Through the combination of face recognition and speaker recognition, it supplies secure interaction. Whenever voice and face-to-face authentication are not possible due to extreme conditions, it creates a backup option that the user can use instead. There is no internet connection required to use this app.

2.5.2.1 AppLock (Android)

Face tracker software AppLock is one of the top free face recognition apps that makes sure that only the user has access to social media apps, financial accounts, and personal information.

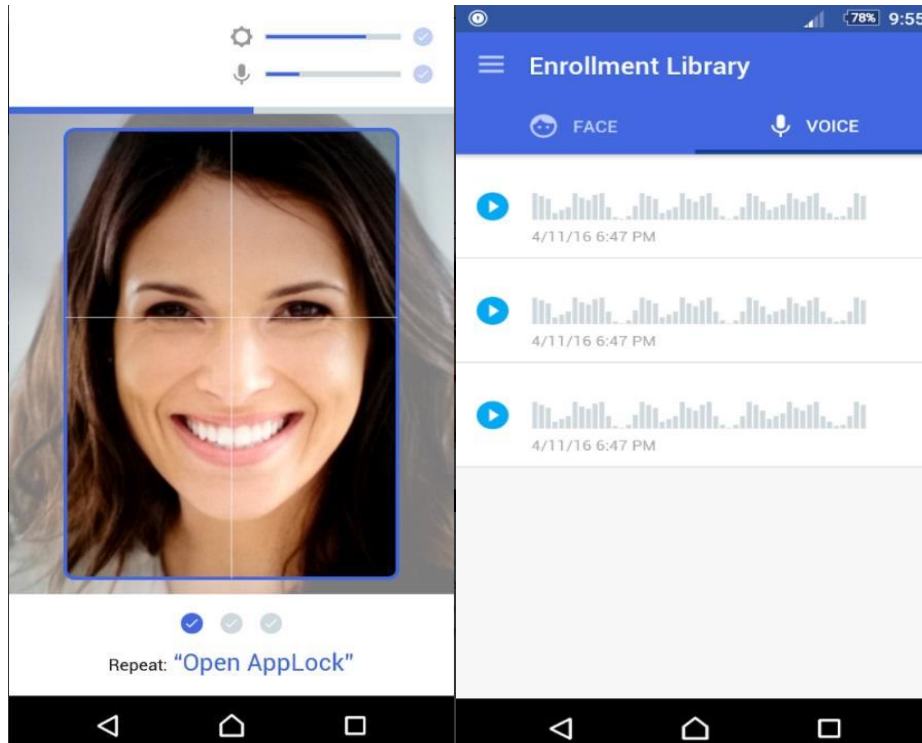


Figure 2 Applock (from spaceo.ca, 2021)

2.5.3 Smarter advertising

By making accurate guesses regarding people's age and gender, face recognition has the potential to make advertising more targeted. Companies like Tesco are already planning to put screens with built-in face recognition at gas stations.

2.5.4 Find missing person

Face recognition can be used to track missing children and human trafficking victims. If missing people are entered into a database, law enforcement can be notified if they are recognised by facial recognition in a public place, such as an airport, retail store, or other public location (FaceFirst, 2019).

2.5.4.1 Identifying people in public

2.5.4.1.1 IFalcon Face Control Mobile by NNTC

An 8-megapixel camera is embedded in the AR glasses' frame, allowing wearers to scan faces in a crowd and compare them with pictures stored in a database. Positive matches are displayed on the see-through display embedded in the glasses. These facial algorithms can recognise an individual in less than a second and can detect up to fifteen faces per frame. They rank in the top three for accuracy in vendors' tests.



Figure 3 ifalcon face control mobile (from spaceo.ca, 2021)

2.5.4.1.2 FindFace

The FindFace face recognition technology was developed by the Russian company NtechLab, a leader in neural network tools. FindFace offers a line of services to the state and to various business sector. (en.wikipedia.org, 2016). A new app that scans a person's face and identifies their identity has sparked fears over public anonymity. A crowd-photography service designed to help users find faces has a 70 percent accuracy rate. In Russia and the former Soviet Union, the app compares photos to social network profile pictures. The social network has more than 200 million accounts (express.co.uk, 2016).

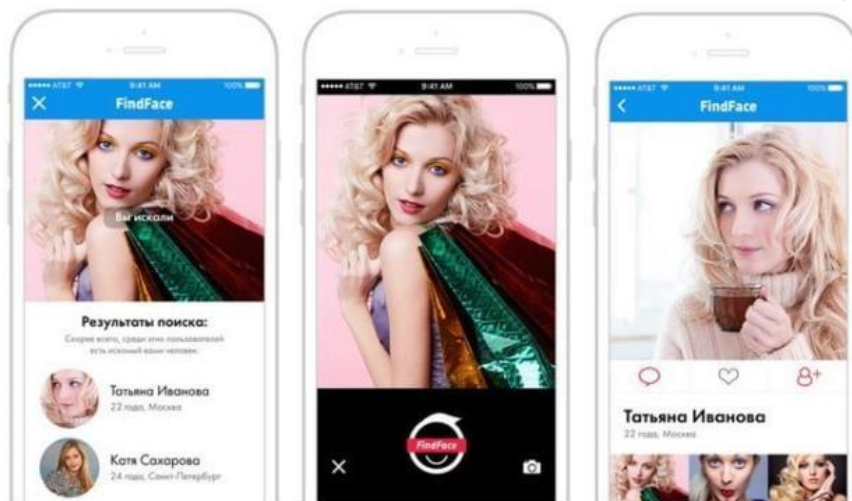


Figure 4, FindFace app (from digitaltrends.com, 2016)

2.5.5 Law enforcement

Face recognition apps, such as FaceFirst's, are already aiding police officers by allowing them to quickly identify people in the field from a safe distance. Before the end of the year, the UK's largest police force plans to enhance its facial recognition capabilities. The Metropolitan Police Service in London will be able to examine old photos from CCTV feeds, social media, and other sources using innovative technologies to track down criminals (Wired,2021). Using facial recognition technology, the Police can use CCTV footage or other images that have become known during criminal investigations to find unknown suspects. The technology checks to see if the image matches one of the photos held by the police of people who have previously been arrested. In London, for the first time, the Police are deploying live facial recognition cameras. It is planned to use the cameras every five to six hours drawing up a list of suspects wanted for serious and violent crimes. An independent review of the cameras has revealed that most matches produced by them are unfounded (BBC News, 2020). The system helps law enforcement agencies, and some businesses find suspects and victims by matching photos and videos against databases such as driver's license files. Nonetheless, civil liberties groups say facial recognition undermines privacy, reinforces racism, and can be misused. Increasing concerns about the misuse of the technology by governments, police, and others led Facebook to shut down its face recognition system and remove more than one billion people's face prints (MSN News,2021).

2.5.5.1 China Police using Face recognition mounted on sunglasses

Chinese police use face-recognition technology to track down and arrest a suspect in a concert crowd of sixty thousand people. Due to facial recognition cameras, the system flagged up the suspect, even though there were thousands of people around and police arrested the suspect on the spot. A camera module is attached to the glasses, which communicate with a smartphone running a police app. When the app executes, it will process pictures of suspects' faces and return information like their names, ethnicity, and gender (The Sun,2018). In addition to focusing on individuals on government deny lists, around forty local governments also use its CCTV recognition to monitor festival attendees and airport travellers (South China Morning Post,2018).



Figure 5, Chinese Police Face Recognition device with sunglasses. (From South China Morning Post,2018)

2.5.6 Attendance System

Face recognition has the potential to track students' attendance in addition to making schools safer. Tablets are being used to scan students' faces and verify their IDs by comparing their images to a database (FaceFirst, 2019)

2.5.7 Facilitate secure transactions

Ant Financial is a Chinese financial services firm that allows users to pay for meals by scanning their faces. Customers put their orders using a digital menu and then pay with a facial scan. They can then purchase their meal after entering their phone number.

2.5.8 Social-media

When Facebook members appear in images, Facebook employs face recognition technology to instantly recognise them. Consequently, it makes it easier for people to find themselves in photos, and it allows them to suggest the tagging of certain persons.

2.5.8.1 FaceApp (iOS, Android)

It is a facial recognition app that was originally launched in 2017 for iOS phones only, but later it was released for Android phones as well. It identifies (facial identification) celebrities and allows users to share photos old and new in the same way other media apps do. It features Hollywood filter selfie technology. The smile filter provides the user to choose the size of a smile. The user can also make a smile toothed or remove teeth. The app allows the hairstyle filter to experiment with different interesting hairstyles, and make the hair blond-haired person, white, black, or brunette, and see how you will look. And other filters such as Gender-swap filter, Age filter, Skin tone lightning filter.



Figure 6, Face app for iPhone and android (from spaceo.ca, 2021)

2.6 Face Recognition Techniques

2.6.1 Face recognition based on Convolutional Neural Networks

CNNs are made up of filters, kernels, or neurons with programmable weights, parameters, and biases. Each filter takes a set of inputs, conducts convolution, and optionally adds a non-linear filter to the combination. The architecture of a convolutional neural network is hierarchical. Equation 1, (from arxiv.org, 2016) shows that the simulation begins with a signal x , and each subsequent step x_j is computed as

$$x_j = \rho W_j x_{j-1}$$

Equation 1, (from arxiv.org, 2016)

In Equation 2, (from arxiv.org, 2016), W_j is a linear operator, while ρ is a non-linearity. W_j in a CNN is usually a convolution, and ρ is a rectifier $\max(x, 0)$ or sigmoid $1/(1 + \exp(-x))$. The layers are filtering whose sum is the convolution of the previous layer.

$$x_j(u, k_j) = \rho \left(\sum_k (x_{j-1}(\cdot, k) * W_{j,k_j}(\cdot, k))(u) \right)$$

Equation 2, (from arxiv.org, 2016)

In Equation 3, (from arxiv.org, 2016) $*$ is the discrete convolution operator:

$$(f * g)(x) = \sum_{u=-\infty}^{\infty} f(u)g(x - u)$$

Equation 3, (from arxiv.org, 2016)

Typically, weights W_j in convolutional neural networks are learned using stochastic gradient descent, utilising the backpropagation algorithm to compute gradients. This creates a highly non-convex optimisation problem.

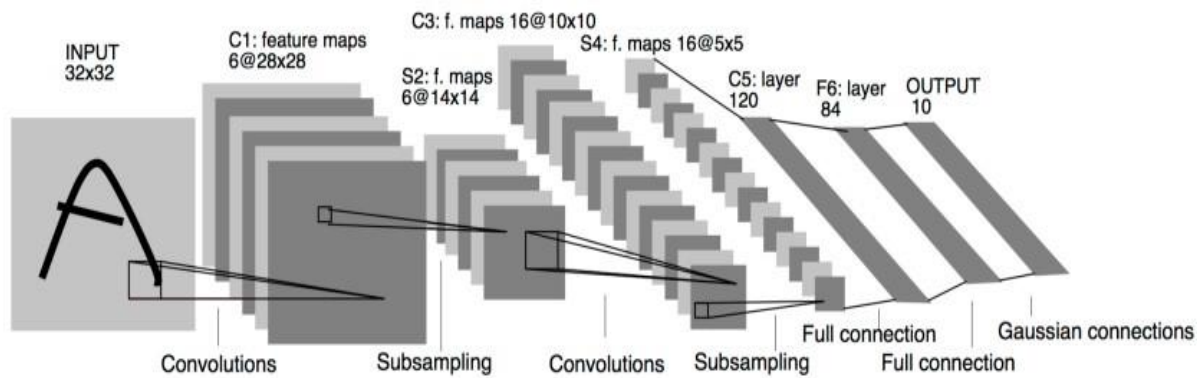


Figure 7, Architecture of a Convolutional Neural Network. (From arxiv.org, 2016)

The structure of CNN contains four varieties of layers such as-

Convolutional Layer: The basic building component of a Convolutional Network, the Convolutional Layer, handles most of the computational significant lifting. By using the layer, features can be extracted from the input data.

Pooling Layer: Each image input is split into separate rectangles that do not overlap with each other. By using a non-linear technique such as average or maximum, each region is down sampled.

Relu Layer: In a Relu Layer, a neuron input is given as x . The rectifier is defined in the literature for neural networks as $f(x) = \max(0, x)$.

Fully Connected Layer: FCLs have each layer of a process connected to every other layer of the process.

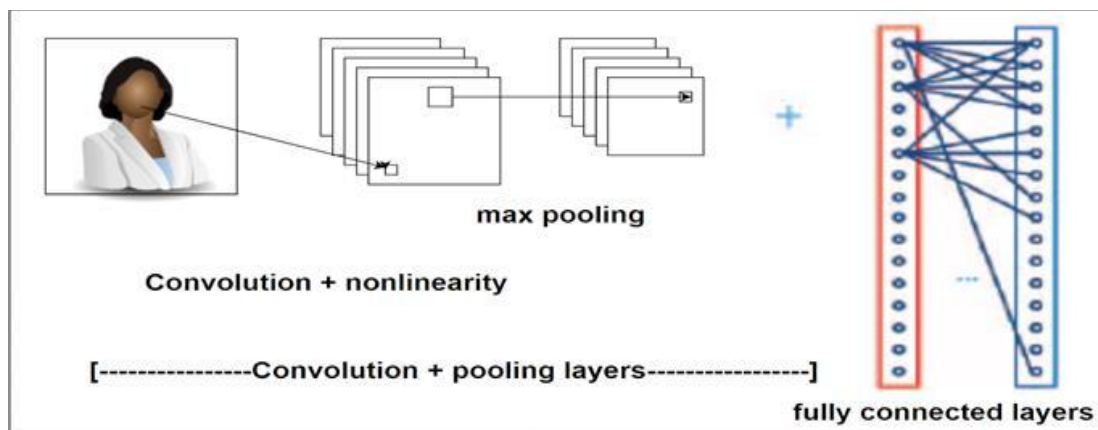


Figure 8, A traditional Convolutional Neural Networks design. (From researchgate.net, 2016)

2.6.2 Holistic Matching

A holistic approach takes the entire face region into account as input data. Examples of holistic methods include Eigenfaces, PCA, Linear Discriminant Analysis, and independent component analysis, among others. The Eigenfaces Method covers face recognition as a two-dimensional recognition problem (q.opengenus.org, 2010).

2.6.3 List of techniques

Feature based (Structural), Model-Based, Hybrid Methods, Viola-Jones, Histogram of Oriented gradients and so forth (q.opengenus.org, 2010).

2.7 Factors affecting accuracy of Face Recognition

The biggest misconception when choosing a face recognition system is that the quality of the system depends directly upon the choice of an FRT algorithm such as Appearance-based, Feature-based etc. A top-class neural network is not enough to produce error-free results automatically. This is demonstrated in “Proceedings of ICRIC 2019” by the fact that high-quality recognition depends on both intrinsic and extrinsic factors.

2.7.1 Ageing

An ageing person may no longer be recognisable to an FRT algorithm as wrinkles appear and facial shapes change. As an individual's skin texture changes, the facial highlights shift. (ijamtes.org, 2018)

2.7.2 Facial expression

A small shift in facial features may cause the neural network to become confused. A person's face typically changes shape when people smile, laugh, or cry. Despite changing facial expressions, researchers continue to develop algorithms for emotion recognition and human identification.

2.7.3 Plastic Surgery

A surgical procedure can alter any part of an individual's face and is a common way of subjectively improving one's appearance. Despite the popularity of plastic surgery, FRT algorithms are unable to recognise people after undergoing such modifications. No neural network exists that can detect a person after severe reconstructive surgery.

2.7.4 Occlusion

Several factors can contribute to partial obstruction of the face, such as a medical mask, sunglasses, specs, earrings, or scarves. Hair, moustaches, and beards may also contribute to partial obstruction. Researchers suggest a variety of methods to help solve these problems. Most of the existing FRT algorithms fail when occlusion occurs. (ijamtes.org, 2018)

2.7.5 Low Resolution

The low-resolution images often used in investigations come from surveillance cameras. High-resolution databases can be helpful in identifying individuals caught in these images.

2.7.6 Noise

Image processing is prone to noise. The most common types of image processing noise include Gaussian, Poisson, Speckle, and Salt and Pepper noises.

2.7.7 Illumination

Due to how light and shadow fall on a human face in varying lighting conditions, images taken under varying lighting conditions may appear confusing to an FRT algorithm. Modern algorithms are susceptible to illumination distortion, but there are ways to compensate for its effect to achieve accurate facial recognition.

2.7.8 Pose variation

In modern databases, non-frontal individual features are not stored, so FRT achieves the best results when the face is viewed from the front. All other poses harm the accuracy of face recognition. FRT is currently dealing with a wide variety of position issues. Suggestions for dealing with them and improving the algorithms are being discussed (ijamtes.org, 2018).

2.8 Limitations of current face recognition technology

2.8.1 Poor Image Quality

Face-recognition algorithms perform better when the image quality is better. However, video scanning has a much lower quality than digital cameras. HD video is only 1080p, and most commonly it is 720p. These values are equivalent to 2 and 0.9 megapixels, respectively, while an inexpensive digital camera achieves fifteen megapixels.

2.8.2 Small Image Sizes

An algorithm designed to detect faces in an image or a still from a video capture is made more effective by comparing the relative size of the face with the enrolled image size. In addition, the small size of the image and the distance from the camera means that the detected face is only one hundred to two hundred pixels in size. Finally, scanning the image for varying face sizes requires a prominent level of processing power. To reduce false positives and accelerate the image processing process, most algorithms allow the specification of a range of face sizes.

2.8.3 Data Processing and Storage

The resolution of high-definition video is quite low, but it occupies a substantial amount of disk space in comparison to digital camera images. Typically, only a fraction of video frames is processed by an automatic recognition system, and agencies use networks of computers to minimise total processing time. The addition of computers requires significant data transfer over a network, which can be constrained by input-output limitations, further slowing down processing speed.

3.Application Requirements and Design Specifications

A brief discussion of the initial application requirements and its design is provided in this chapter. This document outlines the requirements, the design of the solution in the form of UML diagrams, database design and planning of the implementation of actual product.

3.1 System Description

The goal of this project is to build a face recognition system called " faceRek" out of a zeroCam attached to a Raspberry Pi mounted on sunglasses that will run a face recognition system which switches on the Green LED and plays the audio of the celebrity's name, and a website will display the matched picture to the user including a different web page with all the saved pictures in the database. This project will entirely be dependent on Amazon Web Services. The serverless technology will be used to power the facial recognition system. According to the initial specifications, visitors of the website should be able to examine images with a high degree of resemblance, followed by the names of celebrities in the database. For example, with the face recognition system built in this project, the user will need to wear sunglasses, and once the face recognition process (analysing faces) is complete, the user will be able to examine the results on the website by executing the software.

3.2 Stakeholders

The project will be developed for the CST3990 Undergraduate Individual Project module, so its stakeholders are me and my family friends who will chose to use the website. Stakeholders in a real-world commercial environment include website and system owners, developers, and clients.

3.3 Functional Requirements

The requirements from user's perspective and system's perspective in order to use the face recognition system and website.

3.3.1 The User's perspective

- Use the device
 - ✓ Wear the sunglasses with face recognition system.
- Open the website.

3.3.2 The System's perspective

- Code to click pictures with any camera module.
- Code to upload images to AWS S3.
- Save the pictures clicked by the user into S3 Bucket.
- Analyse the pictures using lambda function.
- Use **Amazon Rekognition** to check similarity between Uploaded pictures and the dataset.
- Send the image with higher similarity into database and another S3 Bucket.
- Send message to all clients whenever there is new data in the S3 Bucket using **Websocket API**.
- Send message to all clients whenever the new data is saved into DynamoDB using **REST API**.
- Display names on the web page.
- Display images on the web page.

3.3 Use Case Diagram

The user will be wearing sunglasses equipped with a face recognition system. The user must gain access to the website, which contains display images and display names.

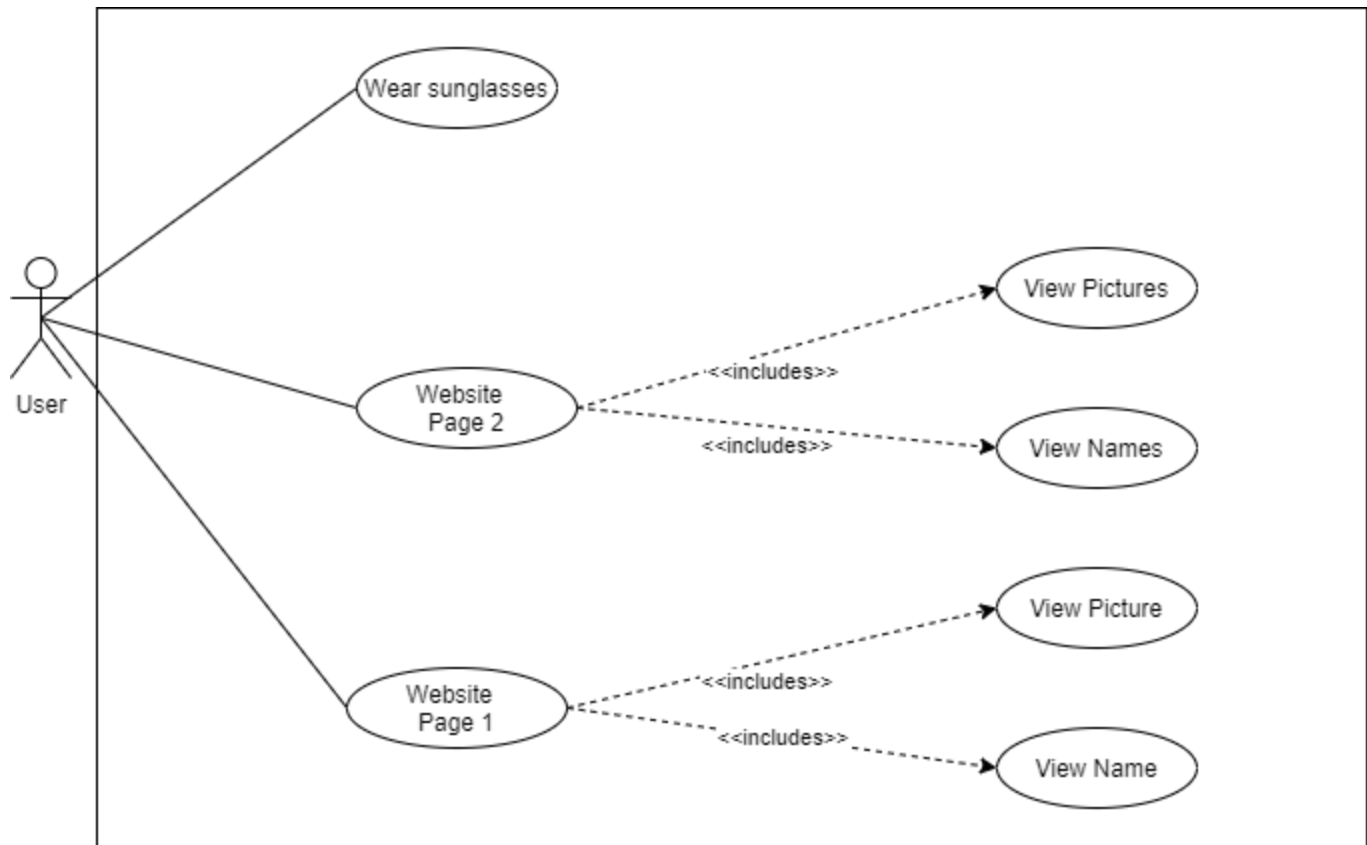


Figure 9, Use Case Diagram

3.4 Activity Diagram

The activity diagram depicts the activities that the user and the system will carry out.

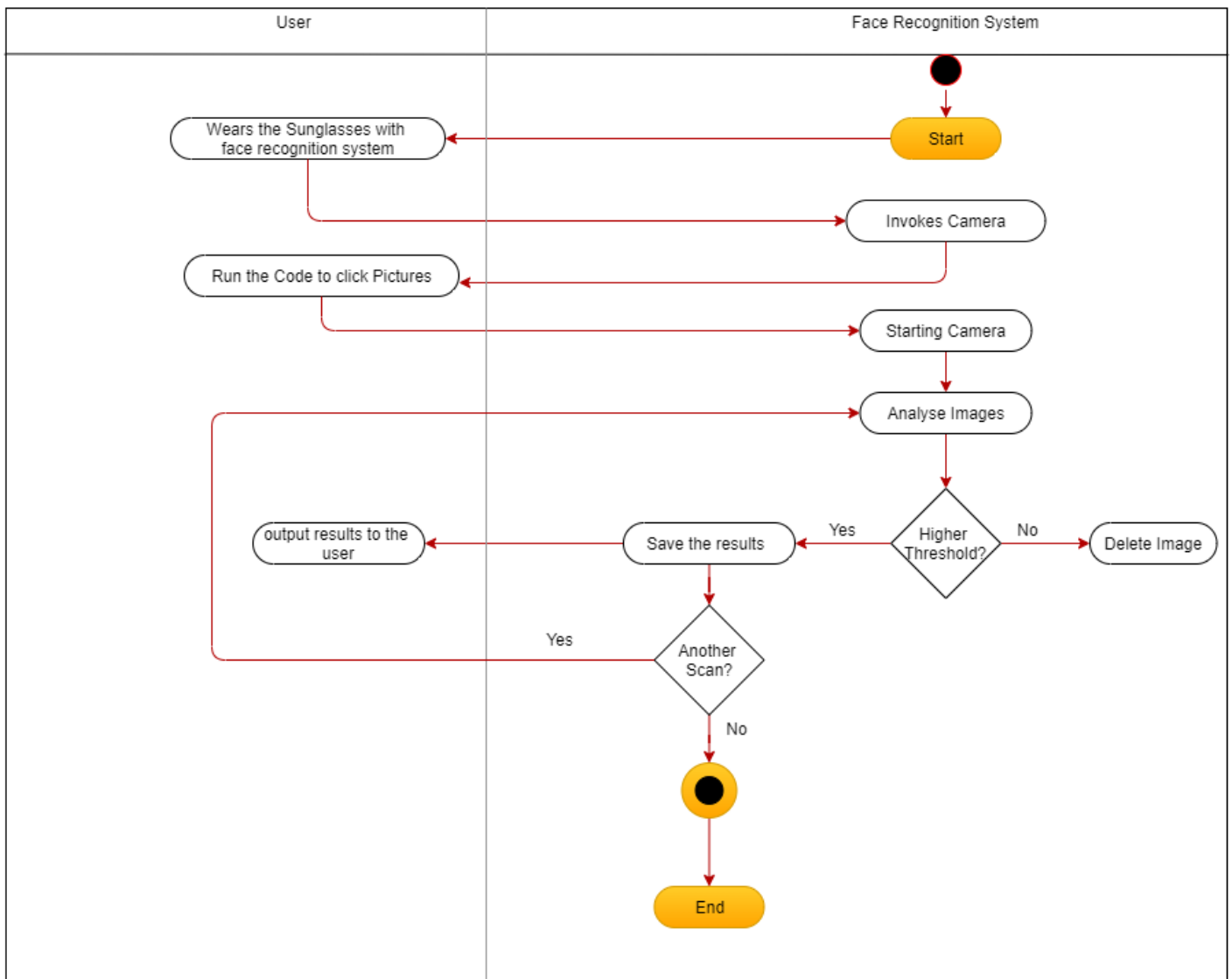


Figure 10, Activity Diagram

3.6 Data Flow Diagram-

The data flow diagram depicts the flow of how the system will operate.

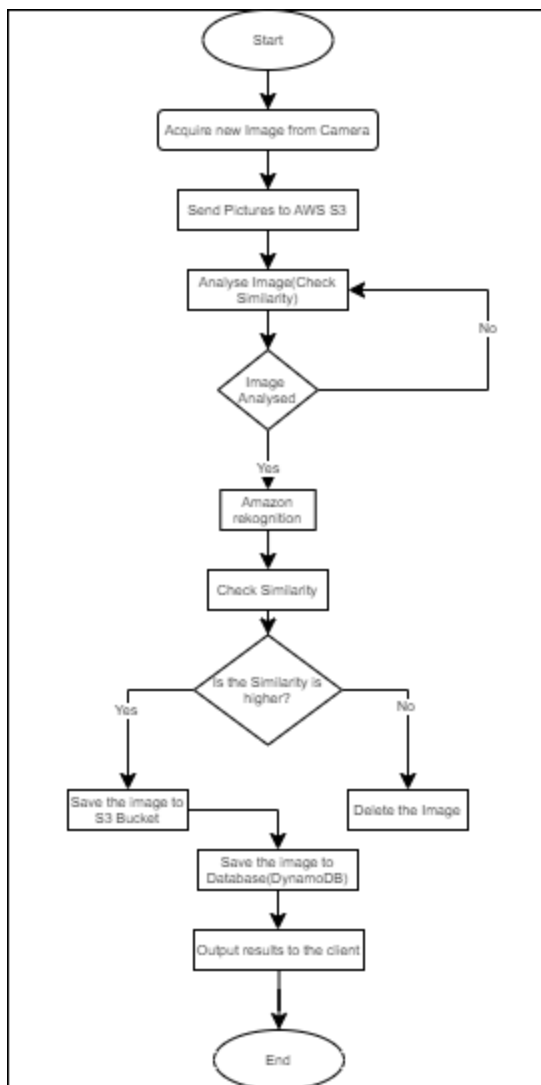


Figure 11, Data flow Diagram

3.7 Database Design

This project will use an Amazon Web Services DynamoDB database with the three tables shown below to carry out this project.

CelebrityImagesWithS3Links			
id	string		
celebrityName	string		
pcitureUrl	string		
Add field			

Figure 12, DynamoDB Table

There are three columns in the **CelebrityImagesWithS3Links** table: the ID, celebrity name and picture URL. This table will be used to store the name and the link to the image with higher threshold which will be stored in S3 Bucket after Amazon Rekognition process.





WebSocketClients			
id	string		
 Add field			

Figure 13, DynamoDB Table

The **WebSocketClients** table holds ID. This table will be used to store IDs of the clients. When a new client connects to the websocket API, this table will help to store the id of that specific client.









CelebrityImageWithOneLink			
Id	integer		
CelebrityName	string		
PictureUrl	string		
 Add field			

Figure 14, DynamoDB Table

There will be three columns in the **CelebrityImagesWithOneLink** table: the ID, celebrity name and picture URL. This table will be used to store the name and the link to the image with higher threshold which will be stored in S3 Bucket after Amazon Rekognition process. This table stores the item with same Id and whenever the latest item will be added to the database it will automatically removes the old item from the database and adds the new one.

3.8 Hardware

This part of the report details the procedures involved in putting the hardware for this project together. It also discusses the hardware implementation step by step.

3.8.1 Raspberry pi with laptop

In this project, the raspberry pi will be used as hardware, connected to the zeroCam to take images. The raspberry pi image is downloaded to the SD card, along with all the contents in the boot directory. "ssh pi@raspberrypi.local" on the console may be used with the ssh file, and the wpa-suppllicant file is used to connect the raspberry pi to wi-fi. The network name and password for the wi-fi connection are stored in the wpa-suppllicant file. VNC Viewer is being installed on the laptop to provide access to the images.

```
pi@raspberrypi:~$ ls
Bookshelf  config  credentials  Desktop  Documents  Downloads  hello  Music  Pictures  Public
Templates  Videos
```

Figure 15, Raspberry Pi Terminal

This system will be conducted using the Raspberry Pi 2 W. This will hold the SD Card with the operating system.

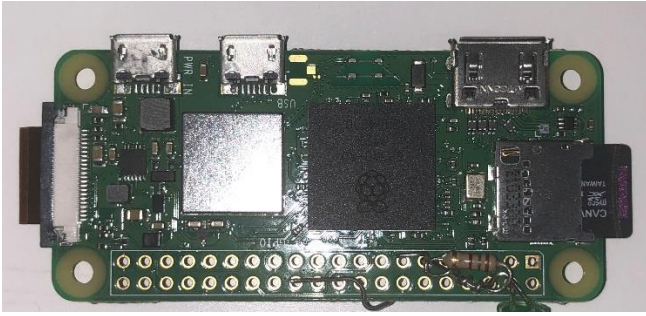


Figure 16, Raspberry Pi 2 W

3.8.2 Camera attached to Raspberry pi

The camera will be connected to the Raspberry Pi in order to take the photos. The code to take a photo with this camera will be included in the raspberry pi's python file. To start the facial recognition system, the user will have to execute this file.

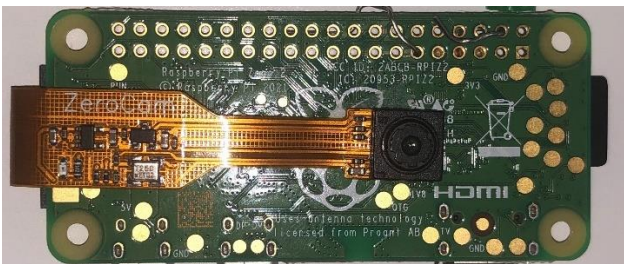


Figure 17, zeroCam connected to Raspberry Pi

This is an example of a photograph taken with zeroCam.

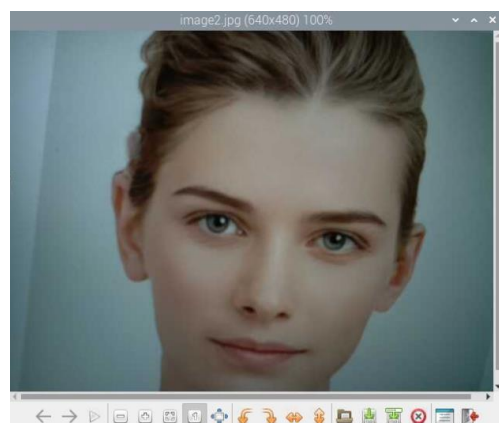


Figure 18, Picture taken by the zeroCam

3.8.3 Uploading Pictures from the Raspberry pi to AWS S3 Bucket

The Python PiCamera module will be used to upload the images taken by the zeroCam. In S3, an unfilled bucket will have been established to hold the photographs uploaded by pi. The python file supplies the code for directly uploading images to AWS S3 Bucket. When the user will take a photo with the camera, this code immediately uploads it to S3. This code will upload images using unique names, allowing it to upload more than one image.

3.8.4 Green LED connected to Raspberry Pi

The raspberry pi will be connected to a green LED. When the raspberry pi gets a message from AWS API Gateway (WebSocket API), the green light flashes.



Figure 19, Green LED attached to raspberry pi

3.8.5 USB to Audio connected to Speaker for Audio Output

The USB to Audio adapter will be linked to the raspberry pi and it will support the transmission of audio from the raspberry pi to the user. The speaker will be attached to the raspberry pi through USB to Audio adapter.

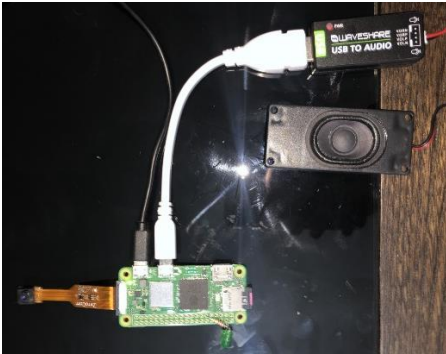


Figure 20, USB to Audio and Speaker attached to raspberry pi

3.8.6 Text to Speech

To convert text-to-speech, the Python Library pyttsx3 will be used. This python code will have a WebSocket client that calls the pyttsx3.init() method to obtain a reference to a pyttsx3. When the client receives a message from the server, it turns it to speech.

3.8.7 Raspberry pi and camera mounted on sunglasses

Finally, the raspberry pi and the zeroCam will be mounted on sunglasses. The user would wear these sunglasses in order to use the facial recognition technology.

3.9 Wireframes

In addition to the hardware notification system, there will be a web page with photos of celebrities. This part holds the website that has been set up to assist the facial recognition technology.

This is the front end of the website. The website splits into two sections.

3.9.1 WebPage 1(View all the recognised Images)

This web page will list all the celebrities discovered previously in the dynamoDB database. The user will be able to view all the images in the database from this page. The Rest API link will be used to access all the photos from the database.

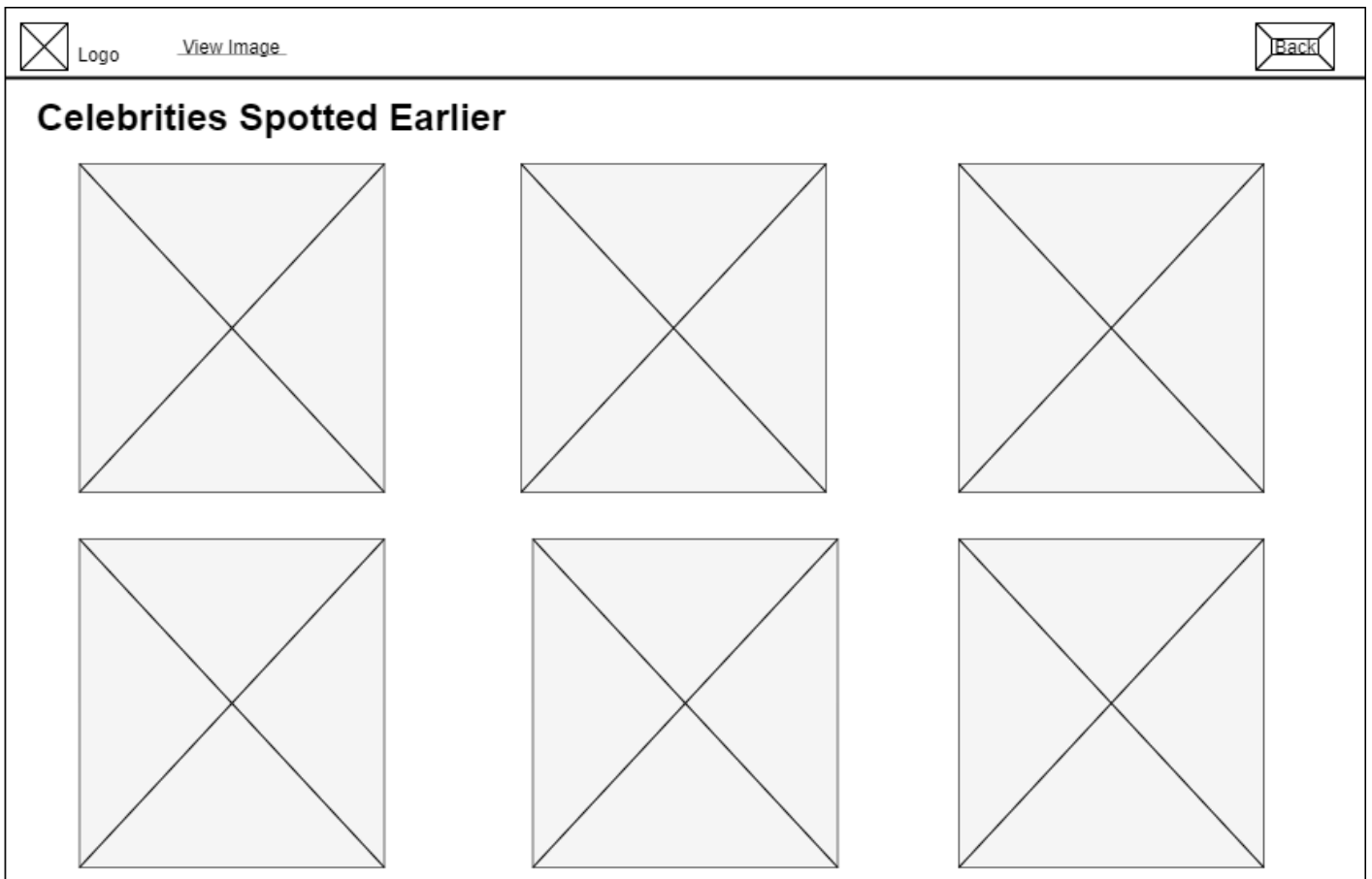


Figure 21, Web Page with all DynamoDB pictures

3.9.2 Webpage 2 (View new recognised Image)

This website's page will reveal the celebrity who is currently being observed. The user will be able to see the celebrity's name and photo. This web page will obtain the celebrity's name using a WebSocket API link and the image URL via a Rest API link.

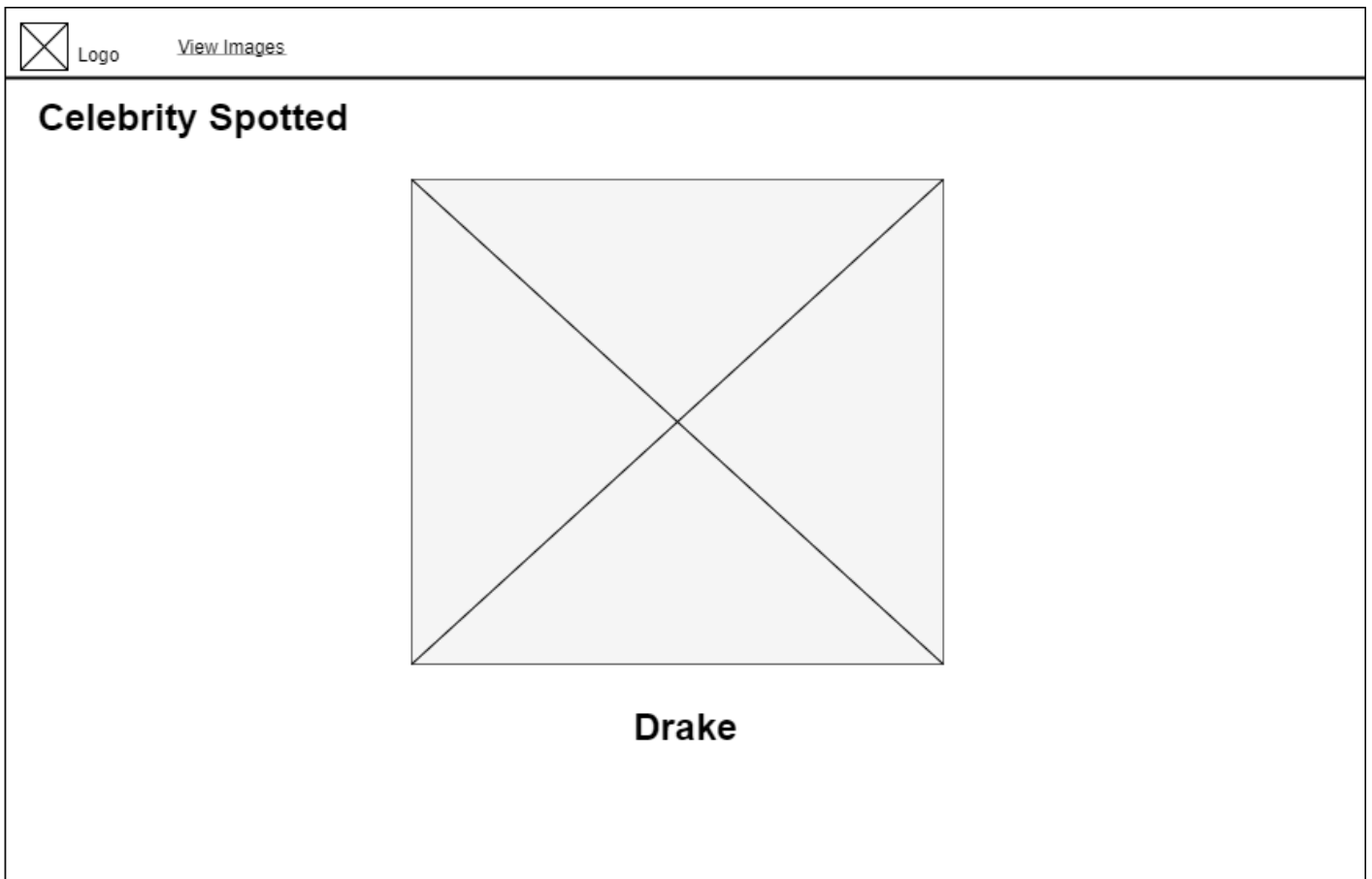


Figure 22, Web Page displaying recognised celebrity

3.10 API Design

To exchange data between the client and the server, the WebSocket API and Rest API will be used with AWS API Gateway.

3.10.1 WebSocket API

To send the new message to the client, the websocket API will be created. In this API, the route 'sendMessage' will be configured, which will be linked to the lambda function. The lambda function will call S3 Bucket, which will retrieve the event's data. So, whenever there will be a new data in the event, this function will retrieve it and sends it to all clients connected to the websocket.

3.10.2 Rest API

There will be a REST API on AWS API Gateway that will be used to retrieve celebrity images and names from AWS dynamoDB and display them on the front end to the user via the REST API link.

4. Implementation

4.1 Overview

To address all the criteria and design specifications specified in the preceding sections of this report, a fully working web application and system software was developed. The facial recognition is powered by a Raspberry Pi, which turns on green LED whenever a photo is found that matches one in the database. In addition to the hardware for this project, there is a web application that shows the current photo of a celebrity as well as earlier pictures saved in order to prove to the client that the facial recognition system is working properly.

4.2 Final Product

This is the ultimate result when all the processes have been completed. The camera is attached to the sunglasses and powers a facial recognition system, as illustrated in the image below.



Figure 23, Final Face Recognition Product

4.3 API Gateway

To send data to the client, the API Gateway is used. WebSocket API and REST API have been implemented in API Gateway.

4.3.1 WebSocket API

The WebSocket API is designed to send a celebrity name to the client. The routes have been configured in the WebSocket API, which is linked to AWS lambda functions in order to retrieve data from events.

4.3.1.1 Routes

To build the connections of API Gateway, four routes have been implemented **connect**, **disconnect**, **default** and **sendMessage**. All routes are linked to lambda functions. The connect route will connect the websocket client to API Gateway and saves the id of client into DynamoDB table. The disconnect route will disconnect the websocket client from API Gateway and delete the client id from the database. The default route just contains response and the sendMessage route will send the message to the client when it is connected to the websocket.

4.3.1.2 WebSocket Connection Link

<wss://9ekmfzaz8b.execute-api.us-east-1.amazonaws.com/prod>

The domain of the link is [wss://9ekmfzaz8b.execute-api.us-east-1.amazonaws.com](https://9ekmfzaz8b.execute-api.us-east-1.amazonaws.com) and stage is [prod](#).

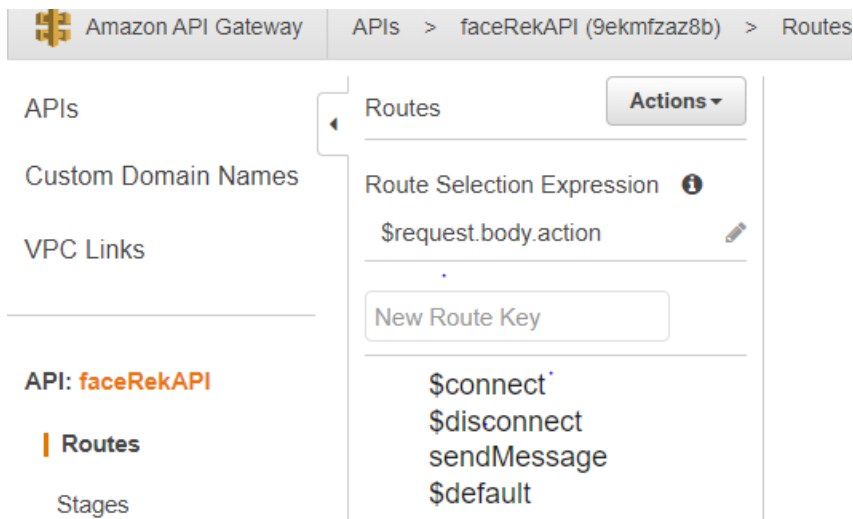


Figure 24, WebSocket API with routes

4.3.2 REST APIs

There are two resources in the Rest API. The first method retrieves all things from the dynamoDB database, whereas the second retrieves one item at a time. GET techniques can be found in both resources. The GET methods are associated with the different lambda functions. The resource **/images** are linked to the lambda function, which returns to the client connected to the website a json response with an array of objects. The second resource **/image** returns to the client one item from the database.

HTTP Method	Resource	Description
GET	/images	returns the array of items from database
GET	/image	returns one item from database

Figure 25, Table showing methods used in REST API

4.3.2.1 REST APIs Connection Link

<https://pdyqa3ex25.execute-api.us-east-1.amazonaws.com/prod>

Where domain is <https://pdyqa3ex25.execute-api.us-east-1.amazonaws.com/> and stage is [prod](#).

To get new Image- <https://pdyqa3ex25.execute-api.us-east-1.amazonaws.com/prod/image>
To get all Images- <https://pdyqa3ex25.execute-api.us-east-1.amazonaws.com/prod/images>

4.4 System Architecture

The serverless architecture used in the development of this web application and hardware for face recognition system is heavily dependent on Amazon Web Services. This type of technology is mainly used because it reduces architecture costs, is more scalable and eliminates the need to manage and maintain servers manually. The architecture used to implement this system is event-based, and individual components of the application are independent of one another. In order to make this operate, several different Amazon web services were used including-

- AWS Lambda
- AWS S3(Simple Cloud Storage)
- DynamoDB
- API Gateway,
- Amazon Rekognition

The web application was built with a variety of technologies. The frontend of the website was built using HTML5, CSS3 and Bootstrap. The **fetch** method in JavaScript is used to request celebrity names and celebrity images from the server, and then load the data from the server on the webpage. The backend was built using a combination of Python and NodeJS. Python was mostly used to take photographs with the zerocam and to connect to Amazon web services by uploading images to the S3 Bucket. As a result, it was used to connect websocket clients to the server via API Gateway through a websocket API connection. It was also utilised to turn on the Raspberry Pi's Green LED, and the Python library pytsx3 was used to broadcast the celebrity's name to the client over the WebSocket API which transmits the name through the speaker. For AWS Lambda functions and the Rest API, JavaScript was utilised.

- The zeroCam is connected to a Raspberry Pi, which uses Python to take photographs.
- The images are then saved with unique names and sent to the AWS S3 Bucket.
- The S3 bucket triggers the lambda function, which runs the facial recognition system. Every time a new image is added to the S3 Bucket, the identical lambda function is invoked. The data for this lambda function is obtained from the event. This lambda function pulls data from two buckets and uses **AWS Rekognition** to find the image with higher similarity.
- When a user uploads an image to S3, it compares it to an image in another S3 bucket, which is the actual dataset for face recognition system.
- If the similarity threshold is more than 95 percent, the image is saved with a unique ID in another S3 Bucket (This bucket holds all the pictures with higher similarity). It also preserves the URLs to the images and the names of celebrities in dynamoDB.
- Two dynamoDB tables: first table to store all the image links and names with unique IDs that it can store more than one image. The second table is used to hold one URL of the picture at a time with the same Id, dropping the old item from the database whenever a new item is added to the database. The lambda function is linked to the dynamoDB database, which has an array of photo URLs with celebrity names. As a json response, this lambda function delivers an array of image URLs. This lambda function is associated with API Gateway (Rest API).
- In the Rest API, the resource **/images** hold the GET method, which invokes the lambda function, which returns an array of picture URLs to the client. When the client connects to the website it displays all the saved pictures with celebrity names from the database. The lambda function sends the response in JSON format.
- To deliver new data to the client, the same API Gateway (Rest API) is employed with different resource. This API is constrained to the lambda function. This API holds a resource named **/image** that takes data from the lambda function (one item at a time) and sends it to the client. In this scenario, the data is shown to the client using REST API Link. The lambda function is also connected to the dynamoDB database with a single image URL. This lambda function returns the new image link that was just added to the database as a JSON response.

Additionally, The S3 Bucket holding photographs with a higher threshold activates a lambda function, which transmits the celebrity's name to the client over API Gateway (WebSocket API).

- This API has a **sendMessage** route that is linked to this lambda function (which supplies the celebrity's name) as a json response. The data for this lambda function is obtained from the event. It only stores the image in this bucket if the similarity is greater than the threshold. The client IDs have been saved in the database, and when the customer leaves, the client ID will be removed from the database. The websocket API communicates with the client directly via the website.

- In order to obtain the data, the raspberry pi also runs the websocket client. In this perspective, there are two websocket clients, one of which is a raspberry pi, and the other is a direct user. However, it transmits the identical information to both clients. Eventually, if there is a message from the websocket server, the raspberry pi switches on the Green LED and sends the name of the celebrity (which is the server's response) to the client through audio output.

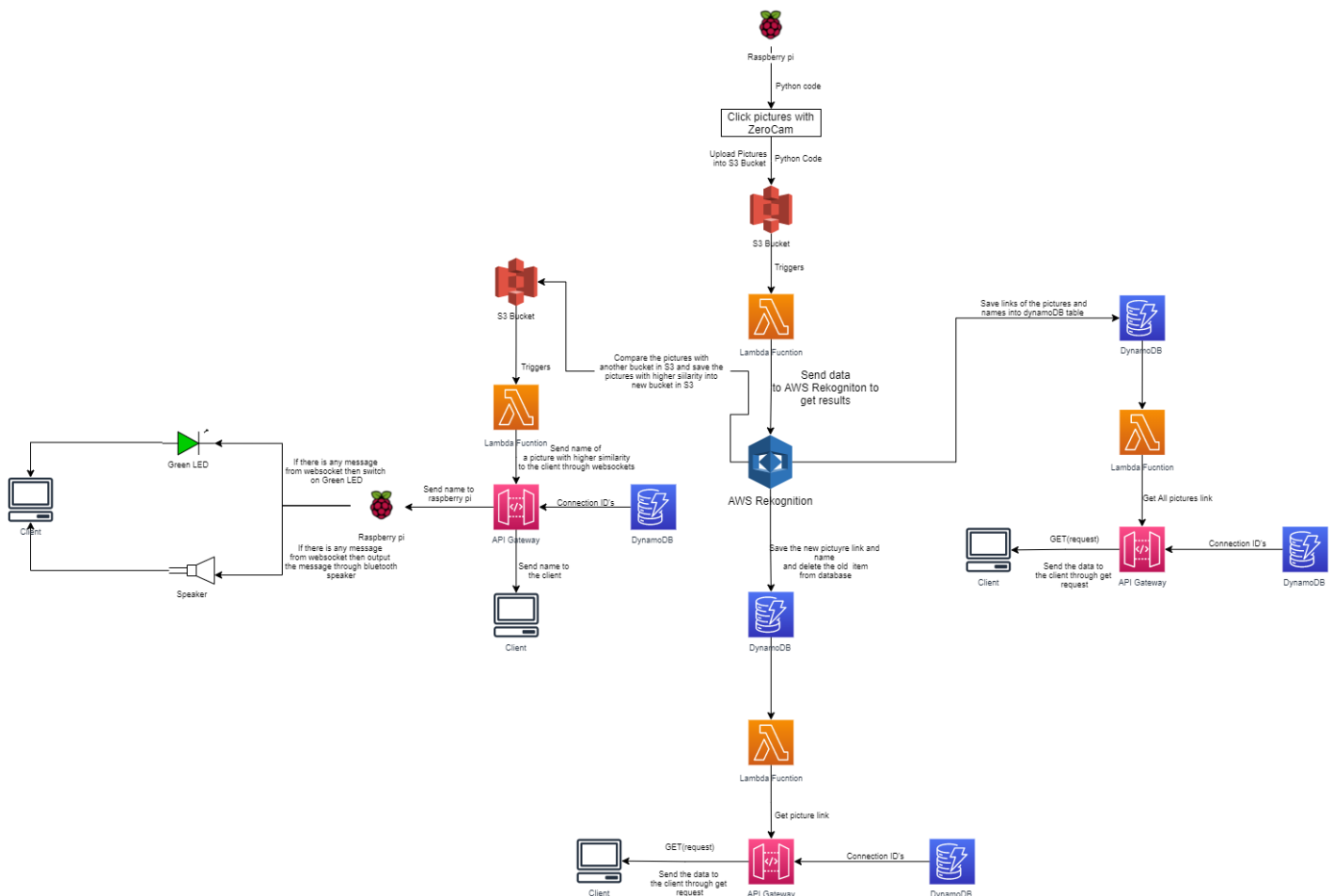


Figure 26, AWS Architecture

4.5 Hardware Description

This project is split into two sections: hardware and software. This section goes into detail about the hardware.

The Raspberry Pi 2 W and zeroCam are used as part of the hardware for this project. The zeroCam is primarily used to take photographs. Below is a description of the hardware's role in this project.

- The Raspberry Pi is used to run the python code that takes pictures with the camera, which uses the python library PiCamera and saves the images in a single folder with unique names. All the images are in jpeg format.
- The Raspberry Pi is linked to the Amazon Cloud services via the .aws folder, which contains the AWS credentials (API Keys). The Python code is used to upload the folder's saved images to Amazon Cloud Storage.
- The next step is to analyse the images in the cloud and sends the results to the client via raspberry pi. The python file contains the WebSocket client, which connects to the WebSocket server via the boto3 library.
- When the client receives data, it turns on a green LED that is linked to the raspberry pi via GPIO pin numbers.
- [[SPEAKERS]]

4.6 Application Description

This is the web application's front-end. The user interaction begins with accessing the page that displays the newly sent image, and the other link on this page takes the user to a different portion of the website.

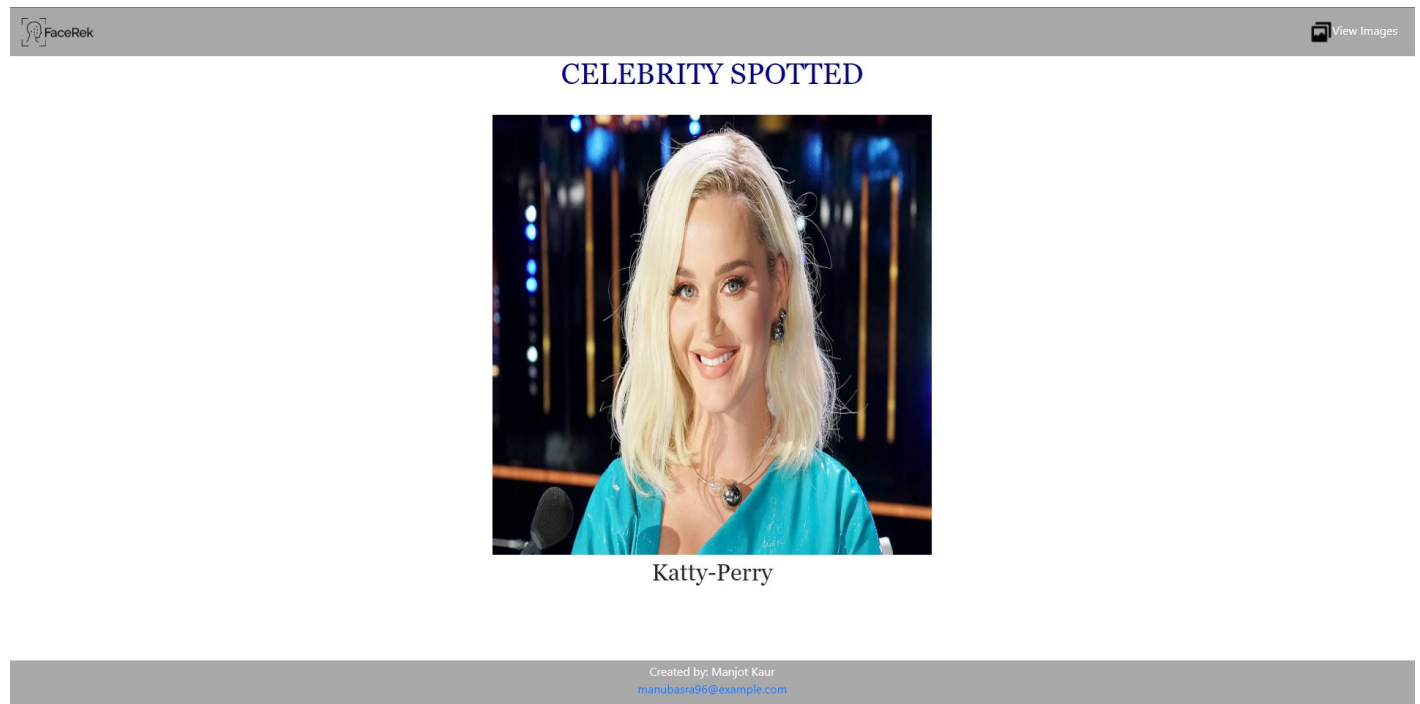


Figure 27, Web Page with one recognised Image

The second page of the web application shows the client previously spotted images that have been recorded in the database.



Figure 28, Web Page with all recognised Images

4.7 Amazon Rekognition

The crucial stage in the facial recognition process was to compare the faces, which was accomplished using **AWS Rekognition** and the **compareFaces** operation. Rekognition is a strong and sophisticated Machine

Learning Vision API that runs on AWS infrastructure. It enables users to do face detection, face recognition, and a variety of other visual processing fast and accurately. To use this API, a Rekognition Collection was initially formed by manually uploading the celebrity dataset. Once a collection was produced, it was compared to the photographs that the user had contributed, and then the user images were obtained from the S3 bucket where the users had uploaded their images at the start of the process. AWS Lambda was used for the recognition process. The lambda function produces a unique uuid for each face recognised in an image and saves it in a DynamoDB database and an S3 bucket.

The **similarityThreshold** parameter was used in the request to establish the least degree of confidence in the match to return in the response. In order to compare the photos, the source image and target image were employed in addition to similarityThreshold. The source image parameter searches the items in S3 Bucket that the user has uploaded, and the target image searches the items in S3 Bucket that is the dataset (including celebrity photographs) for the AWS Rekognition to compare faces. As a result of the multiple target pictures, the source photos are compared to all the images in the target image. If the picture is compared and the similarity is greater than the similarityThreshold, the response is an array of FaceMatches; otherwise, the response is an array of UnmatchedFaces.

```
{
  "SourceImageFace": {
    "BoundingBox": {
      "Width": 0.36378616094589233,
      "Height": 0.9149734973907471,
      "Left": 0.32084253430366516,
      "Top": -0.03116077184677124
    },
    "Confidence": 99.99898529052734
  },
  "FaceMatches": [
    {
      "Similarity": 98.0570297241211,
      "Face": {
        "BoundingBox": {
          "Width": 0.24242399632930756,
          "Height": 0.49609193205833435,
          "Left": 0.39486607909202576,
          "Top": 0.1885744035243988
        },
        "Confidence": 99.99987030029297,
        "Landmarks": [
          {
            "Type": "eyeLeft",
            "X": 0.4450594484806061,
            "Y": 0.39707958698272705
          },
          {
            "Type": "eyeRight",
            "X": 0.5445657968521118,
            "Y": 0.38655444979667664
          },
          {
            "Type": "mouthLeft",
            "X": 0.4629526436328888,
            "Y": 0.5478189587593079
          },
          {
            "Type": "mouthRight",
            "X": 0.5461044311523438,
            "Y": 0.5395722985267639
          },
          {
            "Type": "nose",
            "X": 0.48486828804016113,
            "Y": 0.4818750023841858
          }
        ]
      }
    ]
  },
  "Pose": {
    "Roll": -6.64436674118042,
    "Yaw": -11.74334716796875,
    "Pitch": 0.1795676670351115
  }
}
```

Figure 29, AWS Rekognition response with FaceMatches

```
{
  "SourceImageFace": {
    "BoundingBox": {
      "Width": 0.36378616094589233,
      "Height": 0.9149734973907471,
      "Left": 0.32084253430366516,
      "Top": -0.03116077184677124
    },
    "Confidence": 99.99898529052734
  },
  "FaceMatches": [],
  "UnmatchedFaces": [
    {
      "BoundingBox": {
        "Width": 0.4243519902229309,
        "Height": 0.5225026607513428,
        "Left": 0.2907790243625641,
        "Top": 0.18431076407432556
      },
      "Confidence": 99.99897766113281,
      "Landmarks": [
        {
          "Type": "eyeLeft",
          "X": 0.3918096125125885,
          "Y": 0.40533164143562317
        },
        {
          "Type": "eyeRight",
          "X": 0.5858272314071655,
          "Y": 0.4049634039402008
        },
        {
          "Type": "mouthLeft",
          "X": 0.412102609872818,
          "Y": 0.5798051357269287
        },
        {
          "Type": "mouthRight",
          "X": 0.5743427276611328,
          "Y": 0.5797926187515259
        },
        {
          "Type": "nose",
          "X": 0.47704246640205383,
          "Y": 0.5118825435638428
        }
      ]
    },
    {
      "Pose": {
        "Roll": -1.0892677307128906,
        "Yaw": -4.702332019805908,
        "Pitch": -0.6780520081520081
      }
    }
  ]
}
```

Figure 30, AWS Rekognition Response with UnmatchedFaces

5.Evaluation and Testing

This section includes all the testing techniques.

5.1 Extensive review

This section of the report will analyse the testing stage of the project and gauge how well the system functions according to the previously detailed requirements and design specifications. Especially when working with large scale systems in a commercial environment, testing and evaluation are an integral part of every software development workflow. Distinct types of testing have been performed in order to validate the system.

5.2 UI Testing

5.2.1 HTML Validation

In order to ensure that the website works with different browsers, HTML Validation was used to verify that there were no significant issues in the front end. Nu Html Checker has been used to evaluate the index.html file which is the primary file for this project. This assures that there are no problems associated when showing the front-end to the user. This, in turn, is an important approach to boost user engagement and experience on the website.

Nu Html Checker

This tool is an ongoing experiment in better HTML checking, and its behavior remains subject to change

Showing results for uploaded file index.html

Checker Input

Show

☐ source

☐ outline

☐ image report

Options...

Check by

file upload

Choose file

No file chosen

Uploaded files with .xhtml or .xht extensions are parsed using the XML parser.

Check

Document checking completed. No errors or warnings to show.

Used the HTML parser.

Total execution time 9 milliseconds.

Figure 31, HTML Validation

5.2.2 CSS Validation

CSS validation was conducted to confirm that there were no semantic and syntactic inconsistencies in the web application's frontend code. The CSS file was confirmed using the W3C CSS Validation Service. To evaluate, the file comprising styles was uploaded to this checker.



Figure 32, CSS Validation

5.3 Unit and API Testing

This section describes the methods used to evaluate Rest API and AWS Lambda services.

5.3.1 Lambda Functions Evaluation

While this project employed a serverless approach, most of the interactions between different components of the web application were event-based. This mostly entailed leveraging API Gateway and Lambdas to activate various functionalities. As a result, it was critical to separately evaluate each function and API before merging them into the overall program. Integrating all these various components without evaluating them might lead the system to crash, making troubleshooting extremely difficult. As a result, each Lambda function was evaluated individually on the AWS Console using the testing interface available. In order to test the functionality, S3 events were utilised in the test event.

5.3.2 API Gateway testing Interface

Rest API was evaluated using the AWS API Gateway testing interface.

The image below depicts the GET method using the resource /image, which returns the image's single URL in the response body.

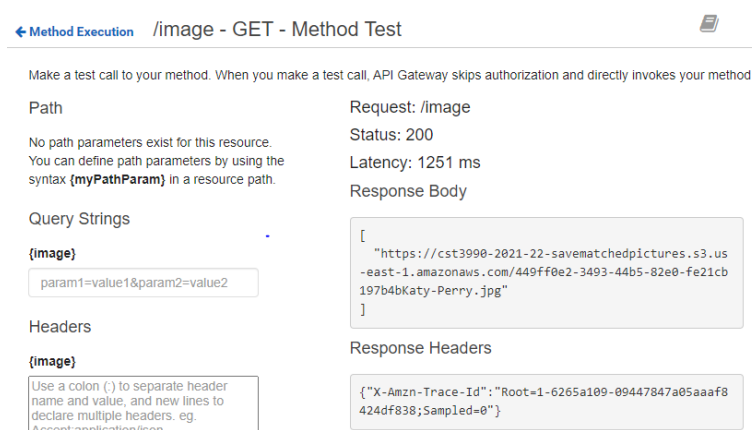


Figure 33, REST API response to get one Image with API Gateway Testing Interface

The image below depicts the GET method using the resource /images, which returns the URLs of all images in the response body.

← Method Execution /images - GET - Method Test

Make a test call to your method. When you make a test call, API Gateway skips authorization and directly invokes your method

Path
No path parameters exist for this resource. You can define path parameters by using the syntax {myPathParam} in a resource path.

Request: /images
Status: 200
Latency: 1186 ms
Response Body

Query Strings
(images)
param1=value1¶m2=value2

Headers
(images)
Use a colon (:) to separate header name and value, and new lines to declare multiple headers. eg. Accept:application/json.

Stage Variables
No stage variables exist for this method.

Client Certificate
No client certificates have been generated.

Request Body
Request Body is not supported for GET methods.

```
[
  "https://cst3990-2021-22-savematchedpictures.s3.us-east-1.amazonaws.com/97bdf81-67b4-40ef-bf90-fca33f2279d4Brad-Pitt.jpg",
  "https://cst3990-2021-22-savematchedpictures.s3.us-east-1.amazonaws.com/27b02849-1e36-420c-acfa-2fd5dd9f80fcJennifer-Lopez.jpg",
  "https://cst3990-2021-22-savematchedpictures.s3.us-east-1.amazonaws.com/0a00b44d-5a61-4c15-aa16-b66fbd4f6671Emma-Stone.jpg",
  "https://cst3990-2021-22-savematchedpictures.s3.us-east-1.amazonaws.com/4c5c68b9-cf0f-4d10-9da1-8de77bf5d5ddDrake.jpg",
  "https://cst3990-2021-22-savematchedpictures.s3.us-east-1.amazonaws.com/449ff0e2-3493-44b5-82e0-fe21cb197b4bKaty-Perry.jpg",
  "https://cst3990-2021-22-savematchedpictures.s3.us-east-1.amazonaws.com/02c38f07-b01a-46a0-9064-401b00b765dcJamie-Foxx.jpg",
  "https://cst3990-2021-22-savematchedpictures.s3.us-east-1.amazonaws.com/81b622ed-545c-4635-b79f-6a380281de2fNatalie-Portman.jpg",
  "https://cst3990-2021-22-savematchedpictures.s3.us-east-1.amazonaws.com/93a4f340-0f42-4133-859b-4c6389f054bdKim-Kardashian.jpg"
]
```

Figure 34, Rest API response to get all Images with API Gateway testing Interface

5.3.3 Rest API evaluate with Postman

Postman was used to ensure that the Rest API ran smoothly and without errors. Both links were evaluated using Postman.

GET https://pdyqa3ex25.execute-api.us-east-1.amazonaws.com/prod/image

Params Authorization Headers (5) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

```
1 [
2   "https://cst3990-2021-22-savematchedpictures.s3.us-east-1.amazonaws.com/449ff0e2-3493-44b5-82e0-fe21cb197b4bKaty-Perry.jpg"
3 ]
```

GET https://pdyqa3ex25.execute-api.us-east-1.amazonaws.com/prod/images

Params Authorization Headers (5) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE	DE
Key	Value	De

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

```
1 [
2   "https://cst3990-2021-22-savematchedpictures.s3.us-east-1.amazonaws.com/97bdf81-67b4-40ef-bf90-fca33f2279d4Brad-Pitt.jpg",
3   "https://cst3990-2021-22-savematchedpictures.s3.us-east-1.amazonaws.com/27b02849-1e36-420c-acfa-2fd5dd9f80fcJennifer-Lopez.jpg",
4   "https://cst3990-2021-22-savematchedpictures.s3.us-east-1.amazonaws.com/0a00b44d-5a61-4c15-aa16-b66fbd4f6671Emma-Stone.jpg",
5   "https://cst3990-2021-22-savematchedpictures.s3.us-east-1.amazonaws.com/4c5c68b9-cf0f-4d10-9da1-8de77bf5d5ddDrake.jpg",
6   "https://cst3990-2021-22-savematchedpictures.s3.us-east-1.amazonaws.com/449ff0e2-3493-44b5-82e0-fe21cb197b4bKaty-Perry.jpg",
7   "https://cst3990-2021-22-savematchedpictures.s3.us-east-1.amazonaws.com/02c38f07-b01a-46a0-9064-401b00b765dcJamie-Foxx.jpg",
8   "https://cst3990-2021-22-savematchedpictures.s3.us-east-1.amazonaws.com/81b622ed-545c-4635-b79f-6a380281de2fNatalie-Portman.jpg",
9   "https://cst3990-2021-22-savematchedpictures.s3.us-east-1.amazonaws.com/93a4f340-0f42-4133-859b-4c6389f054bdKim-Kardashian.jpg"
10 ]
```

Figure 36, REST API GET method to get all Images with Postman

5.3.4 Mocha and Chai Testing

Mocha and Chai testing libraries were used to evaluate the Rest API. These libraries were used to assess the GET methods for getting all the pictures and getting one image.

```
Rest APIs
#getImage
  ✓ should return the one item (image) from the database
#getImages
  ✓ should return the array of images with celebrity names from the database

2 passing (43ms)
```

Figure 37, Results of mocha and chai testing

5.4 Evaluation against requirements

This section compares the initial functional requirements gathered during the project's preliminary stages to the final implemented web application and hardware. The most important parts of the requirements specification for face recognition system can be classified into four categories: taking pictures with camera, uploading pictures to S3 Cloud, comparing faces using AWS Rekognition, turning on Green LED, Audio Output. The most important parts of a web application requirements specification, on the other hand, can be divided into two sections: displaying pictures to the user via Rest API on the web page, displaying celebrity names to the user through WebSocket API. Each of these core elements will be examined separately.

5.4.1 Hardware Evaluation

To begin with, the **first** task is to take pictures with the zeroCam, which allows the user to take pictures of their surroundings. This was completed and implemented as planned by using python code to take the pictures and connecting the camera to the raspberry pi. In order to test the functionality of the camera, a few images were taken.

The **second** task usually involves uploading the images taken by the camera to the AWS S3 Bucket. By using the Python Code to upload images, this task was implemented. The code will automatically upload images to the AWS S3 Cloud when a new image is clicked by the camera. This step of the implementation works properly and was evaluated by uploading more than a hundred pictures into Amazon S3 Bucket.

The **third** task was to compare faces with AWS Rekognition. A dataset of celebrities was manually uploaded to S3 Bucket in order to compare the user's uploaded picture with this dataset. To evaluate the compare faces operation provided by AWS Rekognition, images were uploaded to the cloud one at a time and sent to AWS Rekognition via lambda function, generating the results FaceMatches and UnmatchedFaces. In order to test the functionality, a number of images were compared.

The **fourth** step is to turn on the green LED. This functionality has been completed and evaluated numerous times by sending a message to the Raspberry Pi via API Gateway, which then executes the code to turn on the LED.

The **fifth** task is to play the audio of the message that was delivered to the client via API Gateway.

Reminiscing on the project, successful hardware for a face recognition system was created that met almost all the functional requirements that were established at the outset.

5.4.1 Web Application Evaluation

To evaluate the web application, the **first** step was to display the celebrity images to the user. This was achieved by using the API Gateway (Rest API). It was evaluated several times by sending images from the Rest API link. It is an indication that the functionality of displaying images with API is working correctly.

Additionally, the **second** task was to display celebrity names via the WebSocket API. This procedure was tested by repeatedly uploading the image to AWS S3, which then triggers the lambda function, which communicates the celebrity's name to the client using the WebSocket API. As a result, it was demonstrated that the data sent to the client is correct and works appropriately.

In retrospect, the project resulted in the creation of a successful web application that met nearly all the functional requirements that were established at the beginning.

However, due to limited time, there are some limitations to this project that could be greatly improved in the future. For instance – the hardware for this project can use speakers instead of HDMI cable for the audio output. (See [7.3 Limitations](#)).

5.5 Text-to-Speech Testing

The audio output of the message via WebSocket API (which provides the user with the Celebrity name) is obtained using the python library pyttsx3. The python code was written to make this functionality work. This Python code was tested using a variety of text inputs.

5.6 Systematic Testing

5.6.1 Evaluation with Single Face Image clicked by zeroCam

In order to conduct systematic testing, the zeroCam captured images of six celebrities and uploaded them to Amazon Cloud Storage. These photos only show single celebrities' faces in the image captured by the zeroCam. One image is uploaded to S3 at a time and compared to the dataset's 10 to 20 images. The image was clicked from this angle and compared to the actual images in the database.

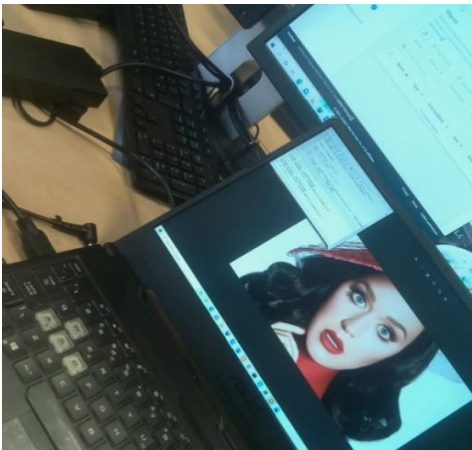


Figure 38, Picture clicked by zeroCam



Figure 38, Original Image in database

The outcomes of this testing are shown below.

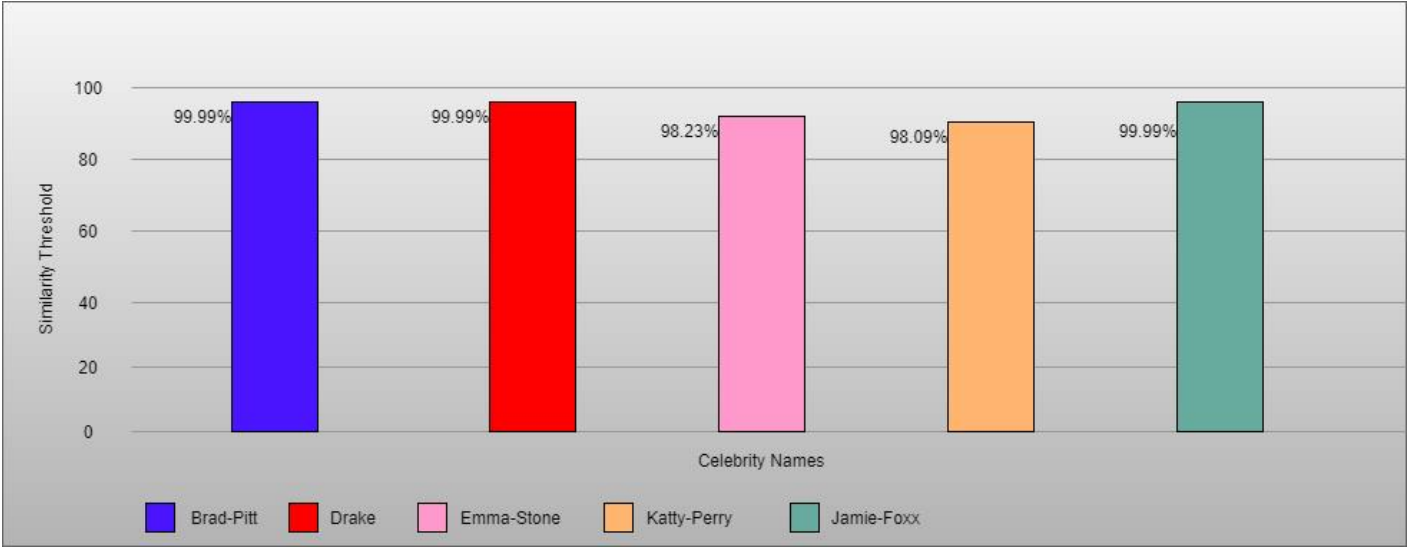


Figure 39, Results of face Recognition

5.6.2 Evaluation Using Photographs of Celebrities in the Crowd

In order to conduct this evaluation, the pictures of celebrities in crowd were taken with zeroCam. Which compared to the images in database.



Figure 40, Celebrity in crowd



Figure 41, Original Image in database

The results of this testing are shown below.

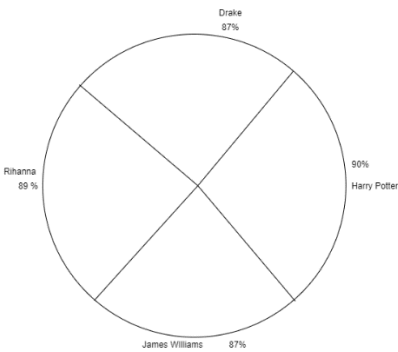


Figure 42, Results of the testing

5.7 Field Testing

This project's field testing was done to ensure the functionality of the face recognition system, hardware, and web application. During the field testing, a celebrity's image was pasted on a tree to evaluate the system in the wild. During the viva, a video demonstration of field testing will be shown.

6. Benefits

The following are the most significant advantages of using faceRek:

1. The system makes it easier for people to locate family members, friends, and loved ones in crowded places.
2. Using this system, people can locate celebrities in crowded areas.
3. Blind people can use this system without difficulty to recognise people.
4. This system can be used for security purposes for people with visual impairment, for example, in the event of a crime.
5. By wearing these faceRek sunglasses in the opposite direction, people can easily determine who is behind them.

7. Conclusion

The goal of this project was to create hardware for a face recognition system as well as a web application that can help a user find a celebrity in crowded places. In order to implement this system, the project investigated a few technologies that work well together. The primary focus was on the field of face recognition and the issues it raises. This chapter describes in detail each of the system's major technologies and summarises their technological advancements and limitations.

7.1 Ethical Reflections

As a result, this project is being implemented in the CST3990 Individual Project. Because of ethical concerns, the face recognition system cannot be used in public. For example, if the task is to take a photo of a random person, some people may object to offering their faces and storing their images in a database.

7.2 Cloud Provider

Deploying an application to the cloud is fantastic and has a plethora of advantages. Getting to know and use several Amazon services had a significant impact on this project. Due to a lack of knowledge about the cloud environment when beginning the development of the face recognition system using Amazon Rekognition, some minor errors and poor technological choices were made. For example - The AWS lambda functions can trigger dynamoDB events while sending one image to the client, allowing the client to receive new data every time. This will make it a lot easier to notify client with new data.

7.3 Limitations

In final analysis, the project resulted in the creation of a successful web application that met almost all the functional requirements that were established at the start of the project. However, due to time constraints and unforeseen circumstances, there are some limitations to this project. Some of these issues, as well as potential solutions, are discussed below.

- **Hardware-** It can take a long time for the Raspberry Pi to upload the image to Amazon Cloud Storage. It can be replaced with a different device that can withstand more strain.
- **Software-** The names of celebrities are obtained via WebSocket API for web applications, and images are obtained via REST API. However, it is also possible to obtain the names using the REST API. This allows the user to view the name for a longer period on the web page. The DynamoDB events can be used instead of scanning the table.

7.4 Future Work

More features can be added in the future to improve this project.

- To implement hardware, two or more cameras can be added to take pictures while running a face recognition system on all of them at the same time.
- Instead of using zeroCam to take images, the python code can be used to play video camera to find people or celebrities.
- To avoid recognising celebrity posters or advertisements in public, the detector can detect the distance between the camera and the person to determine whether there is a person present.
- The camera can give a signal while taking pictures (such as audio, using LED's).
- There is no login or registration for the user, so every user views the same data. The user login and registration can be added to the website in order to display only the images that the user has uploaded rather than all data.

7.5 Summary

In conclusion, because of face recognition, faceRek has a lot of room for development. Most of the requirements described in the design are met; however, there is still plenty of room for improvements and enhancements. Spending more time designing the front-end of the website, for example, would result in greater intuitiveness and ease of navigation. In addition to recognising celebrities and loved ones in public, faceRek has some other advantages. It can assist a blind person in recognising a person from a distance. It can be used for blind people's security so that they can react in accordance with the person they know.

8. References

- "Facebook to shut down Facial Recognition System, Delete 1 billion face Images – MSN News". MSN News, 2021. Web. 11 Mar 2021.
- "How Amazon Rekognition helps in the fight against some of the worst types of crimes – Amazon Staff." Amazon, 2019. Web. 25 July 2019
- "Met Police to deploy facial Recognition cameras – BBC News." BBC News, 2020. Web. 20 Jan 2020
- Bansal, A." The study of factors affecting face recognition." Volume 32, 2019, pp. 669 – 671.
- Burgett, G. "FindFace is a new facial recognition app that could end public privacy." Available at: <https://www.digitaltrends.com/photography/findface-social-networks-detect-people-public-with-70reliability/> (Accessed: 18 May 2016)
- Gallarate, S. (2019). "Chinese Police facial recognition glasses." Available at: <https://www.fairplanet.org/story/chinese-police-officers-are-wearing-facial-recognition-glasses/> (Accessed: 9th July 2019)
- Geralt, J. (2021)." Facial recognition 2021 and beyond- trends and market." Available at: <https://www.iscoop.eu/facial-recognition/> (Accessed: 22 June 2021)
- Javvy, S. "15 of the best face recognition API in 2021: Rec Faces Overview." Available at: <https://recfaces.com/articles/face-recognition-apis> (Accessed: 19 Feb 2021)
- Kasar, M. (2016)." Face Recognition using Neural Networks." Available at: https://www.researchgate.net/profile/ManishaKasar/publication/301727666_Face_Recognition_Using_Neural_Network_A_Review/links/5ef18af5a6fdcc73be96ccc2/Face-Recognition-Using-NeuralNetwork-A-Review.pdf (Accessed 3 Jan 2016)
- Keach, S. (2018)." EYE SPY Chinese police using SMART SUNGLASSES with facial recognition to spot criminals in crowds." Available at: [Chinese police using SMART SUNGLASSES with facial recognition to spot criminals in crowds \(thesun.co.uk\)](http://www.thesun.co.uk/tech/4461231/chinese-police-using-smart-sunglasses-with-facial-recognition-to-spot-criminals-in-crowds/) (Accessed: 7 Feb 2018)
- Koushik, J." Understanding convolutional Neural Networks." Mellon University, 2016, pp. 2- 3
- Lo, K. (2018)." Face Recognition Sunglasses by China police." Available at: <https://www.scmp.com/news/china/society/article/2132395/chinese-police-scan-suspects-using-facial-recognition-glasses> (Accessed 7 Feb 2018)
- Mowat, L. "The end of privacy: New facial recognition app allows users to identify strangers in crowd." Available at: <https://www.express.co.uk/news/world/671258/Facial-recognition-app-Findface-allows-users-to-identify-STRANGERS-in-crowd-privacy-ends> (Accessed: 17 May 2016)
- Nec, (2020)." The brief history of facial Recognition". Available at: <https://www.nec.co.nz/market-leadership/publications-media/a-brief-history-of-facial-recognition/#:~:text=This%20was%20the%20date%20that,marketing%20term%20for%20facial%20recognition.>

- Patel, R. (2021). "10 Best Face Recognition Apps in 2021". Available at: <https://www.spaceo.ca/best-facerecognition-apps/> (Accessed: 12 Jan 2021).
- Raviv, S. "The secret History of facial Recognition." Available at: <https://www.wired.com/story/secrethistory-facial-recognition/> (Accessed: 21 Jan 2020)
- Romans, K. (2019). "Face recognition uses." Available at: [21 Amazing Uses for Face Recognition – Facial Recognition Use Cases \(facefirst.com\)](https://facefirst.com/21-amazing-uses-for-face-recognition-facial-recognition-use-cases/) (Accessed: 23 Jan 2019)
- Roomi Mansoor, M. (2013). "A review of face recognition methods." Available at: <https://www.worldscientific.com/doi/abs/10.1142/S0218001413560053> (Accessed 27 Nov 2013)
- Simonite, T. (2021). "Face recognition is being banned-but it is still everywhere." Available at: <https://www.wired.com/story/face-recognition-banned-but-everywhere/> (Accessed: 22 Dec 2021)
- Stephanie, K. (2021). "Facial recognition issues and problems." Available at: <https://www.thalesgroup.com/en/markets/digital-identity-and-security/government/inspired/facialrecognition-issues> (Accessed: 2 Feb 2021).
- Thakkar, D. "Iris Recognition Scanners vs. Fingerprint Scanners: Compare and Contrast." Available at: <https://www.bayometric.com/iris-recognition-scanners-vs-fingerprint-scanners/> (Accessed: 20 Jan 2019)
- Tolba, A.S. (2006). "Face Recognition: A Literature Review." Available at: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.179.2182&rep=rep1&type=pdf> (Accessed 2 Feb 2006)
- TraiChuk, A. (2021). "Face Recognition for real time." Available at: <https://hackernoon.com/6-best-opensource-projects-for-real-time-face-recognition-vr1w34x5> (Accessed: 28th April 2021)
- Walsh, K. (2020). "How Accurate is Facial Recognition Today?" Available at: <https://recfaces.com/articles/how-accurate-is-facial-recognition> (Accessed: 5 Nov 2020).
- Yashwant, S. (2010). "Various Techniques used for face recognition." Available at: <https://iq.opengenus.org/techniques-for-face-recognition/> (Accessed: 7th Mar 2010)
- Zhang, X. "Pattern Recognition." Cognition, Volume 42, 2009, pp. 2876 – 2896

