CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Biometric Authentication for the Web: A Face Recognition System

Bachelor's thesis in Computer Science and Engineering

Erik Gälldin
Linus Lindström
Maria Logren
Mikael Motin
Emil Svensjö
Tabita Tengblad

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2025

Bachelor's thesis 2025

Biometric Authentication for the Web: A Face Recognition System

Erik Gälldin
Linus Lindström
Maria Logren
Mikael Motin
Emil Svensjö
Tabita Tengblad

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2025

Biometric Authentication for the Web: A Face Recognition System

Erik Gälldin Linus Lindström Maria Logren Mikael Motin Emil Svensjö Tabita Tengblad

Supervisor (Handledare): Syed Umer Bukhari, Department of Computer Science and Engineering
Examiners: Patrik Jansson and Arne Linde, Department of Computer Science and Engineering
Graded by teacher (Rättande lärare): Magnus Almgren, Department of Computer Science and Engineering

Bachelor's Thesis 2025
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2025

Biometric Authentication for the Web: A Face Recognition System

Erik Gälldin, Linus Lindström, Maria Logren, Mikael Motin, Emil Svensjö, Tabita Tengblad

Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

This bachelor thesis presents the design and implementation of a proof of concept web application for biometric authentication using face recognition. The goal was to investigate if face recognition could function as an accurate, user-friendly and secure alternative to password based login systems on web platforms. The project included developing a face recognition pipeline using existing open-source models, with added features such as anti-spoofing and encryption for data protection. The system was implemented as a web application and was evaluated through a set of user tests and performance tests on datasets. The results show that the system achieves a high accuracy and usability, even though spoofing remains as an issue. Future work includes improving the spoofing detection, fine-tuning the models for better generalization and developing the system into a scalable authentication API.

Sammanfattning

Detta kandidatarbete beskriver designen och implementeringen av ett system för biometrisk autentisering via ansiktsigenkänning. Målet var att undersöka om ansiktsigenkänning kan fungera som ett tillförlitligt, användarvänligt och säkert alternativ till lösenord på webbplatser. Projektet omfattade utvecklingen av en ansiktsigenkänningspipeline bestående av open-source-modeller, med tillägg som anti-spoofing och kryptering av användardata. Systemet implementerades som en webbapplikation och utvärderades genom användartester och prestandatester. Resultaten visar att systemet är både användarvänligt och tillförlitligt, även om spoofing-träffsäkerhet kvarstår som ett problem. Exempel på framtida utvecklingsområden är förbättring av spoofingdetektering, att finjustera modellerna för att möjliggöra bättre prestanda samt vidareutveckling av systemet till ett skalbart autentiserings API.

Keywords: face recognition, face detection, face verification, authentication, web security, convolutional neural network, artificial intelligence

Acknowledgements

First and foremost, we would like to thank our supervisor Syed Umer Bukhari for providing insightful advice, relevant reading material, and support throughout the entire project. We also extend our thanks to the supervisors from the department of Communication and Learning in Science for their valuable guidance on improving and structuring the report. Finally, we are grateful to all the individuals who participated in our user testing. Your comments provided valuable feedback on the user experience and helped us improve the application to make it more aesthetically pleasing and intuitive. Thank you!

Erik Gälldin, Linus Lindström, Maria Logren, Mikael Motin,
Emil Svensjö, Tabita Tengblad, Gothenburg, June 2025

Contents

List of Figures

List of Tables

Glossary

Authentication A process of proving your identity before letting you in.

Biometric Physical characteristics used to verify an individual's identity.
Bootstrap A popular CSS framework for developing responsive websites.

Cascading Style Sheets (CSS) A style sheet language used for the presentation of a document written in HTML.
CelebA-Spoof A face anti-spoofing dataset based on the CelebA dataset, containing both genuine and spoofed face images.
Confusion Matrix A table used to describe the performance and precision of a classification model. It uses True Positives, False Positives, True Negatives and False Negatives.
Convolutional Neural Network (CNN) Is a type of deep learning model specifically designed to process structured data, such as images.
Cosine Similarity A way to measure similarity between two vectors, by calculating the cosine of the angle between them.

Deep Learning (DL) A subset of machine learning that uses neural networks with many layers to automatically learn patterns from large amounts of data.
DeepFace A lightweight face recognition and facial attribute analysis framework for Python including multiple state-of-the-art models.

Eigenface A facial recognition method that represents faces as a set of key features, making it efficient for identification.
Euclidean Method A method to calculate the distance of a straight line between two points in a space.

Face Detection (FD) The process of identifying and locating human faces in digital images or video.
Face Identification The process of determining a person's identity from a facial image among multiple possible identities.
Face Recognition (FR) A technology that uses face identification and face verification to identify individuals.
Face Spoofing A biometric attack where a fake face is presented to fool a face recognition system.
Face Verification A biometric process used to confirm a person's identity by comparing their facial features with a stored image.
Feature Extraction The process of transforming raw data into numerical features that can be used for identification.

FEI Face Database A research-purpose database of face images from the University Center of FEI.
Flask A lightweight WSGI web application framework written in Python.

Georgia Tech Face Database A dataset containing webcam-quality images of individuals created at the Georgia Institute of Technology.
Google FaceNet A deep learning model developed by Google for face recognition and verification.

Haar (Haar-like features) A set of digital image features used to detect objects in images, especially faces.
Haar Cascades A machine learning-based approach that uses a cascade of classifiers trained with Haar-like features to detect objects (like faces) quickly and efficiently.
HTML (HyperText Markup Language) The standard language for creating webpages and web applications.

L2 Normalization A method for normalizing a vector to have unit length based on the L2 norm (Euclidean norm).
Labeled Faces in the Wild (LFW) dataset A commonly used dataset of face images used for evaluating the performance of face recognition systems.

Metric Scores Uses F1 Score, Recall, Precision, and Accuracy used to evaluate model performance.
Multi-layer Neural Networks Neural networks with multiple layers between input and output, used in tasks like image and speech recognition.
MVP (Minimum Viable Product) A product with the minimum features necessary to satisfy early adopters and provide feedback for future development.

OAuth 2.0 An industry-standard protocol for authorization.
One-time password (OTP) A temporary password that can only be used once for authentication.
Open Source Software for which the original source code is made openly available and may be redistributed or modified.
OpenCV An open-source computer vision and machine learning software library.

ROC Curves A graphical plot that illustrates the diagnostic ability of a binary classifier system.

SQL Injections A code injection technique that might destroy a database, exploiting vulnerabilities in an application's software.
Supabase An easy to use database that provides a way to save important data.

Tailwind CSS A CSS framework for rapidly building custom user interfaces.
Transfer Learning (TL) A machine learning technique where a pre-trained model is adapted to a new task.

User Experience (UX) The overall experience of a person using a product or service.

User Interface (UI) The point of interaction between a user and a digital system.

YuNet A lightweight, real-time face detection model.

1
Introduction

This introductory chapter outlines the rationale behind the goal of this project: to develop a web-based face recognition authentication system that is accurate, secure, and user-friendly. It also discusses the limitations involved in such a project.

1.1 Background

Today several challenges surround the use of passwords. In Sweden, the number of reported data breaches increased by 36% from 2022 to 2023, and the independent organization Stöldskyddsföreningen is forecasting this trend to continue, largely due to the growing use of social media [1]. This development affects both private individuals and corporations. On a global scale, the average cost of a large data breach reached a record high of 4.88 million dollars in 2024, according to an IBM report, as reported by the online magazine Computer Sweden [2]. Alarmingly, the most commonly stolen data in these breaches was personal identifiable information, and the most common attack vector was stolen or compromised login credentials.

These statistics indicate that passwords are fundamentally flawed as tools for authentication, due to their vulnerability to bad actors. While some risks can be mitigated by creating secure, unique passwords for each service and avoiding password sharing, enforcing such rigorous password management on a large scale is probably unrealistic.

An alternative to passwords is biometric-based authentication. Biometrics are defined as the unique physical or behavioral characteristics of humans, and include fingerprints, iris patterns, facial features and voice patterns. Using biometrics like these for authentication offers several advantages, such as high accuracy, ease of use and uniqueness [3]. Additionally, biometric authentication is physically non-intrusive and eliminates the need for users to manage passwords diligently. While biometric data can still be stolen or leaked and requires sophisticated protection [3], the responsibility for preventing such attacks does not fall on the user.

Face recognition (FR) allows devices, such as computers and smartphones, to identify a person solely based on a camera image or video. Since cameras are common in devices that require authentication, FR can be seamlessly integrated without additional hardware, enabling fast, real-time authentication. Moreover, several modern models achieve high accuracy, for instance, Meta's DeepFace at 97.35% and Google's

FaceNet at over 99%, which can be compared to human accuracy at 97.53% [4]. Combined with the fact that many of these models are open-source, these advantages make FR authentication particularly interesting.

Modern FR research dates back to the early 1990s, and was initially focused on dividing facial images into smaller feature components, known as *Eigenfaces*. Facial classification could then be achieved by analyzing how the position of each Eigenface varied [4]. Similar but more sophisticated approaches created models more insensitive to lighting conditions and different facial expressions. However, performance degraded when images differed significantly from the examples in the dataset used to train the models. These results led to a shift in focus toward so-called *high dimension feature representation*, which, despite outperforming earlier methods, still lacked functionality in real-world scenarios. The real paradigm shift - achieving human-like performance - only occurred with the introduction of Deep Learning for model training. In other words, by using an AI approach with multi-layer neural networks to learn to recognize and generalize patterns of facial characteristics, these models became highly proficient at recognizing faces.

Nowadays, biometric authentication methods such as face or fingerprint recognition are commonly used in smartphones, tablets and laptops (for example, Apple's FaceID and Microsoft's Windows Hello), but are still rarely encountered on the web. Developing an FR-based authentication system as a web application is therefore a relevant project.

1.2 Purpose

The main objective of this project is to explore how biometric authentication can serve as an accurate, secure, and user-friendly alternative to traditional password-based login systems on web platforms. Using face recognition (FR) technology, the project aims to demonstrate the feasibility of integrating such a high-performing system into a website.

To achieve this, a proof-of-concept web application was developed and tested, allowing users to register and authenticate themselves via FR. The system processes facial data using machine learning algorithms to determine access based on a stored biometric database.

1.3 Scope

To ensure that this project remained feasible within the given time frame and available resources, certain constraints were established. These limitations define the project's focus and set clear boundaries for its implementation.

1.3.1 Use of Video-Based Recognition

There are multiple approaches to implementing FR. This project relied on live video footage captured by a device's camera for user authentication, rather than still images. Still images pose a higher security risk, as an unauthorized user could bypass authentication by presenting a photograph of someone else. While live video reduces this risk by making such attacks more difficult, it does not eliminate them entirely. For instance, an attacker could hold up a video or high-quality image in front of the camera. To address this, we explored incorporating a custom FR model trained to detect such spoofing attempts.

Additionally, using live video enhances the user experience by enabling a more seamless and interactive authentication process.

1.3.2 Use of Open-Source Pre-trained Models

This project relied on open-source, pre-trained machine learning models for FR. By utilizing existing tools, higher-quality results are ensured, significant development time is saved, and the limitations imposed by licensing restrictions and the costs of commercial solutions are avoided. While open-source solutions may not be as optimized or feature-rich as proprietary alternatives, they are both practical and cost-effective.

2
Theory

This chapter presents the theoretical foundation for FR. It begins by describing the main steps in a typical pipeline, from face detection to recognition. It then introduces transfer learning as a method for adapting pre-trained models to new tasks. Finally, it discusses key security aspects that affect the reliability of facial recognition-based authentication systems.

2.1 The Face Recognition Pipeline

Systems that perform user authentication on an input image using FR can be divided into three fundamental steps: face detection, feature extraction, and face recognition inference [5][6]. However, before the detection step, the input image must undergo preprocessing. To capture as much facial data as possible, the raw image data is enhanced through methods such as filtering (e.g., applying a grayscale filter) and fitting. Deep Learning models, often with a Convolutional Neural Network (CNN) structure, have proven highly effective for FR tasks and are therefore widely used in FR pipelines [6]. The basic structure for a FR authentication system can be visualized, like in the pipeline in Figure 2.1, and the different steps will be explained further in this section.

2.1.1 Face Detection

Face detection (FD) is the initial step in the FR pipeline and it is necessary to determine whether a human face is present in an image. Challenges that obstruct FD include variations in scale, location and orientation of the face, illumination, facial expression, pose, poor resolution, and occlusion [4]. Several methods and models have been developed to create detectors that achieve good accuracy despite these challenges. According to Kumar et al. FD approaches can be divided into two main categories; feature based and image based approaches [7].

Feature based approaches aim to detect features that are commonly found in a face. According to a definition from TechTarget, the easiest features to detect in a human face are the eyes [8]. However, there are numerous other features that could be searched for, such as common measurements and shapes, skin tone or gray scale based approaches where the facial features are likely to be darker than the rest of the face regardless of skin tone [7]. Feature based FD can also include methods for edge detection, and motion detection if the input image data is in video form. The

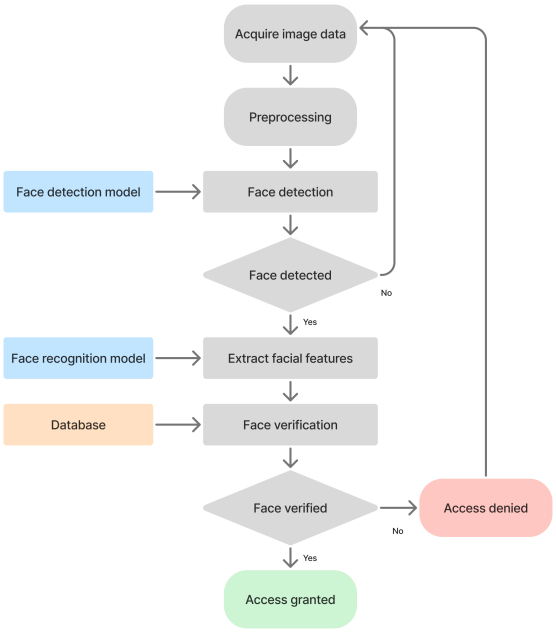

Figure 2.1: Diagram depicting the pipeline for a FR authentication system

feature based methods for FD are easy to implement but can struggle with getting good results on images affected by previously mentioned challenges with variations in images.

Image-based FD include neural networks, linear sub-space methods such as Eigenfaces and statistical methods [7]. Neural networks and most commonly CNNs are used widely today as they achieve a high accuracy despite some of the challenging variations in the image data. CNNs consist of hidden layers that during training develop different methods for finding patterns in the training data [9]. The efficiency and accuracy of a CNN is defined by the number of layers and parameters along with the quality of the training data. Kumar et al. mention that image-based approaches, such as the use of neural networks, for FD can be difficult to implement [7]. However, the research in the area has rapidly become more comprehensive, providing multiple models available by open source. Sheldon et al. states that the usage of CNNs for FD is now the most common approach [8].

2.1.2 Feature Extraction

After the detection step is carried through the next part of the FR pipeline is extracting features from the detected face. During the extraction step, information related to face identification properties, such as the size, shape, and relative distances of the eyes, nose, and mouth, is extracted and stored in a feature vector [5]. According to Wang et al. there are four main stages in the extraction process; filtering, encoding, spatial pooling, and holistic representation [10]. The filtering is applied to define

the wanted features and it can include different levels of complicated filters. The subsequent encoding is used to format the features into a histogram or a feature vector. Spatial pooling is then applied to improve the feature vector with information about where in the image the different features belong. Finally the holistic representation often includes dimensional reduction. This ensures that the output of the feature representation is of a manageable size making the next steps in the FR pipeline feasible with storing and computational matters.

There are many possible ways to perform feature extraction, including the Eigenface analysis mentioned in Section 1.1. In recent years, DL techniques, particularly CNNs, have become increasingly popular for this stage of the pipeline.

2.1.3 Face Recognition Inference

To complete the FR pipeline, a decision must be made on whether the face is recognized based on a comparison of the collected data with database entries. This inference step is application-dependent, with two common approaches: identification and verification [11]. In identification, the extracted feature vector is compared against all feature vectors stored in a database to determine if a match exists, and if so to whom it belongs [4]. This method is commonly used in applications such as surveillance and tracking [12]. In verification, the extracted feature vector is compared to a single pre-registered user's feature vector stored in the database to confirm a match [4]. This is the preferred method for authentication systems, where the user's identity is stated in the login process [13].

Both identification and verification rely on comparing the geometric and structural features stored in the extracted facial feature vectors. While achieving high accuracy is crucial, the system must also allow for variations in factors such as illumination, facial expressions, and slight positional differences between the newly captured image and the stored reference. To accomplish this, a similarity score is computed using a similarity function. Similarity functions used for FR can be based on cosine similarity, statistical or probabilistic models, or other types of logic [14].

The balance between accuracy and adaptability is set by defining an appropriate threshold for the similarity score. A lower threshold allows for greater tolerance to variations but increases the risk of false positives, where an incorrect person is identified or authenticated. A higher threshold reduces false positives but may also lead to false rejections, where a legitimate person is not recognized or authenticated.

2.2 Transfer Learning

Transfer learning (TL) is a technique within machine learning in which a model developed for one task is reused as the starting point for a model on a second, related, task. This is particularly useful because it requires less time and resources. For example, TL can be done with limited access to labeled training data. In the context of deep learning, this typically involves using CNNs that have been pre-trained on

large datasets, and then fine-tuning them for a specific application [15].

The early layers of a CNN typically learn to extract low-level features from the input data, such as simple patterns or structures relevant to the specific domain [16]. In the context of FR, these layers often capture low-level structures such as the contours of the eyes, the outline of the nose, or the shape of the mouth. Since these features are relevant across many visual tasks, the early layers are usually retained when adapting the model to a new domain. Later layers, which focus on more task-specific patterns, are instead retrained or replaced to better fit the new objective.

2.3 Security Aspects

Security is a critical consideration when developing web applications that handle sensitive user data and provide authentication mechanisms. In systems relying on FR, both general web application vulnerabilities and biometric-specific threats must be addressed to ensure robust protection. This section introduces several security aspects relevant to the design and implementation of such systems, including SQL injections, spoofing attacks, and encryption.

2.3.1 SQL Injections

SQL (Standard Query Language) injections can be used to add SQL code to an input field in a web application [17]. They can also be used to gain information about the behavior of the database collected by error messages or analysis of the time consumption for computation after different input attempts. To prevent user input from containing harmful SQL code or minimize its impact on the database there are multiple methods that can be implemented. One option is to limit the allowed input from the user [18]. Some struggles with this prevention method are that depending on the application the user might need to have the possibility to provide free text input and characters that in some cases might be used for harmful injections can in other cases have legitimate function. The access to database specific information, through e.g. error messages, should also be limited to the minimum of what the user needs to prevent further SQL injection attacks.

2.3.2 Spoofing and Anti-Spoofing

Attempting to gain unauthorized access to a system by presenting a photograph, a pre-recorded video, or a 3D model of a person is referred to as spoofing the system [19]. FR systems, particularly those found in smartphones and tablets, often utilize depth-sensing cameras to construct a 3D representation of the user's face, thereby improving their resistance to spoofing attempts [20]. In contrast, most laptop and desktop computers are equipped with standard webcams that lack depth perception, making them more vulnerable to such attacks. To address this limitation, systems can incorporate specialized classifiers trained specifically for face

anti-spoofing, enabling the detection of presentation attacks even in the absence of depth data.

2.3.3 Encryption

Encryption is an essential part of modern security practices and is used to protect sensitive information. By using encryption algorithms, sensitive data can be scrambled into an indecipherable format. Only those authorized with access to a decryption key can unscramble the data back to its original form. As data breaches become increasingly common and regulatory data protection requirements have arisen through standards, encryption has become increasingly necessary in application design [21].

3
Model Selection Process

To establish a foundation for selecting the models for FD and FR to be implemented in the authentication system, numerous publications were reviewed. This chapter describes the selection process and summarizes the chosen models.

3.1 Collection of Sources

Relevant sources for the review were identified using Google and PubMed. Searches were conducted using keywords such as "Face Recognition", "Face Detection", "Face Verification" and "Comparison". Only freely available sources were considered. Each selected source was critically evaluated based on publishing date, author credibility, objectivity, and consistency with other sources to ensure reliability.

Given the rapid advancements in AI and deep learning (DL), face recognition is a fast-evolving field. Although older sources may be less reliable for recent developments, foundational concepts from earlier studies remain relevant.

To ensure sufficient documentation, the most commonly referenced models were compared across multiple studies. The selection of models for this project was based on three key factors; frequency of use in research, computational efficiency and accuracy when tested on the Labeled Faces in the Wild (LFW) dataset. This is the most commonly used dataset for evaluating FR models, but the YouTube Faces Database (YFD) is also frequently cited.

3.2 Chosen Face Detection Model

Based on the reviewed literature, conclusions were drawn regarding the most suitable models for this project. Face detection models need to perform well on varying images as described in Section 2.1.1. There are numerous models available for this, some consisting of more simple algorithms like Haar-cascades and many consisting of CNNs. A comparative study published in 2022 evaluated several models based on speed in Frames Per Second (FPS) and Average Precision (AP) [22]. The results indicated that RetinaFace and YuNet provided the best balance between FPS and AP.

Further testing under varying lighting conditions, facial expressions, poses, occlusion, face scales, and skin tones confirmed the strong performance of these two models [22]. Both are available with open source, which is also a requirement for

this project. RetinaFace has a good reputation but is relatively slow and performs better on smaller faces, which may be problematic when the person is closer to the camera [23]. A difference between the two models is that YuNet requires fewer parameters than RetinaFace, making it more suitable for devices with limited computational power. Face authentication systems would likely be deployed on both mobile devices and computers, making this aspect desirable to ensure compatibility along all used devices. Additionally, YuNet's lower computational demand results in faster performance, improving the user experience. Based on these factors, YuNet was chosen for the initial face detection in this project.

3.3 Chosen Face Recognition Model

For this project's authentication system, an open source FR model was required. Like the development of detection models, advancements in deep learning and CNNs have significantly improved the accuracy of FR systems [4]. Earlier methods can not compete with modern deep learning based approaches. Therefore a model consisting of CNNs would be of highest relevance.

While evaluating the models most frequently referenced in the literature, the Deepface library stood out as a good option. This library includes eight of the highest-performing and most commonly used FR models [24]. Among them, Google FaceNet achieved the highest accuracy score on the LFW dataset, with a reported accuracy of 99.63% [25]. FaceNet is widely recognized as the state-of-the-art face recognition model, making it the preferred choice for authentication in this project.

4

Tools

This chapter describes the tools and models used to develop the web application and build the FR authentication system. The system was implemented using Flask as the web framework and Supabase as the database for managing user data. Several Python libraries and models were integrated to perform tasks such as FD, feature extraction, verification, anti-spoofing, and data encryption. The sections below outline each tool in detail and explain its role in the overall system architecture.

4.1 Flask

Flask is a Python web framework known for its flexibility and lightweight design. As a microframework, it provides only the essential functionality needed to build web applications, allowing developers to customize their projects with additional components as needed. This minimalist approach enables quick development. Flask also provides easy testing and debugging through its built-in development server.

Since the project focused on FR, a lightweight and easy-to-set-up framework was needed to minimize overhead and get the server-side components running quickly. This allowed for more time to test and refine the FR functionality. Additionally, Flask's flexibility made it easy to integrate FD and FR models, as well as various database solutions, making it an ideal choice for the application.

4.2 Supabase

Supabase is an open source relational database built on PostgreSQL that provides developers with a scalable, real-time database. Its built-in features and use of rows and columns make it an easy database to set up and manage. This allows for an advanced database without much prior knowledge. Supabase also supports a range of security features such as row level security (allows you to control and restrict access to individual rows), HTTPS encryption (secures data as it travels between server and client), edge functions security (controls who can access and run server side logic) and protection against SQL injections (through the use of parameterized queries and strict access controls).

For this project, Supabase was chosen due to its intuitive way to setup and manage data, which significantly reduced the time required to implement and made development more efficient. Another reason was that it made collaboration easier, since

all team members could access and work with the same data in real time.

4.3 YuNet

YuNet was used as the FD model. Its role is to locate faces and key landmarks in each video frame before passing the relevant data to the models used for feature extraction and verification. As an open source, real time detector developed by OpenCV, YuNet provides a good combination of speed, accuracy, and ease of integration, making it particularly useful for web applications where both performance and lightweight deployment are important.

4.3.1 Model Integration

YuNet was integrated using OpenCV's face detector API and was configured with a confidence threshold of 0.9, meaning a face is detected if the model is at least 90% confident that a face is depicted. A non-maximum suppression threshold of 0.1 and an input resolution of 320x320 was used. Non-maximum suppression is a step used in object detection to eliminate overlapping face detections by only keeping the one with the highest confidence score, ensuring each face is represented by a single, distinct bounding box, even when multiple detections occur close together.

4.3.2 Detection Pipeline

The face detection process begins by YuNet detecting all present faces in the provided image and then returning both bounding box coordinates and five facial landmarks for each detected face. These landmarks correspond to the positions of the eyes, nose, and corners of the mouth. The detection results were subsequently used for both validation and preprocessing before recognition.

4.3.3 Face Positioning Evaluation

To ensure that only prominent and well positioned faces proceed to the recognition stage, each detected face was evaluated on its bounding box area, as well as with a custom face alignment check using the detected landmarks. The vertical difference between the eyes was normalized by their horizontal distance to measure head tilt, while the vertical difference of the mouth corners was used to check for unusual expressions. Additionally, the horizontal displacement of the nose from the midpoint between the eyes was calculated to determine lateral misalignment. These calculated values were compared against certain thresholds, and if any exceeded their respective limits, the face was considered misaligned and excluded from further processing. This filtering should significantly improve the consistency and accuracy of the face recognition in the later stages of the process.

4.4 FaceNet

In this study, the model used for extracting face features is a 512-dimensional version of FaceNet. This particular version of FaceNet was acquired through DeepFace, an open-source FR framework that provides pre-trained deep-learning models for face analysis. The model converts an image of a face into a feature vector, enabling similarity measurements through Euclidean distance. The following subsections describe the role of FaceNet and the preprocessing steps involved.

4.4.1 Preprocessing for Feature Extraction

Before feature extraction, detected faces were preprocessed to standardize input dimensions and improve recognition accuracy. To ensure the most relevant face was used, the largest detected face in the image was selected. The cropped face was then resized to 160x160 pixels to match the input requirements of FaceNet. Pixel values were normalized between 0 and 1, and a batch dimension was added before being passed into the model. These preprocessing steps hope to ensure consistent input quality, improving the reliability of the extracted facial embeddings.

4.4.2 Feature Extraction and Verification with FaceNet

The FaceNet model maps faces into a high-dimensional feature space that uniquely represents a person's facial identity. The model is trained with a triplet loss function, where a triplet consists of an anchor image of a face, a different image of the same face, and an image of a different face. The loss function is designed to minimize the distance between the faces of the same identity within the feature space and maximize the distance between the different faces. Given an input image of a face, the model extracts a 512-dimensional feature vector as a representation of the scanned face to allow for identity verification through vector comparisons. L2 normalization was then applied to these feature vectors for scale invariance to ensure accurate face comparisons.

The extracted feature vectors were used for face verification by computing the euclidean distance between the new, and already stored, feature vectors. Since the vectors were L2 normalized, the distances could range between 0 and 2. If the distance between two vectors fell below a set threshold of 0.9, the system recognizes the face as a match.

4.5 Anti-Spoofing Model

In order to enhance the security of the face authentication system against spoofing attacks, a dedicated anti-spoofing model was integrated into the system. The model helps distinguish between real, live face inputs and spoofing attempts like pictures or replayed videos. The anti-spoofing functionality is based on an open source project [26], which provides an efficient, lightweight model trained on the CelebA Spoof dataset [27] to recognize common spoofing attempts in realistic environments

with a specified accuracy of 93%. The model is compact and optimized, making it especially suitable for web-based face authentication systems, since speed and user responsiveness is crucial.

4.5.1 Model Integration

The anti-spoofing model is based on a lightweight CNN trained to detect spoofed faces based on subtle differences in image texture and structure. After initialization, the model is fed face crops with a size of 128x128 pixels, with values normalized in a range of 0 to 1. Predictions are made by running the processed image through the model and applying a softmax function to the output to derive the real class probabilities. A prediction is considered real if the probability of the "real" class exceeds a threshold of 0.98.

4.5.2 Spoof-Detection Pipeline

The anti-spoofing module was used after a face has been detected by YuNet and cropped from the live webcam feed. Once the face is processed and passed through the anti-spoofing model, only inputs classified as real faces are passed forward to the feature extraction and verification stages using FaceNet. If a spoofing attempt is detected, the authentication process is stopped, preventing further processing.

4.6 AES-GCM Encryption

Advanced Encryption Standard (AES) developed by the National Institute of Standards and Technology is a symmetric key encryption algorithm used to store secure data by converting it into an unintelligible form called ciphertext. To decrypt the encrypted data into its original form called plaintext, an AES key is required. Its symmetric nature means that the same key is required for both encryption and decryption [28]. Its efficiency and security have led to widespread adoption, including the Apple ecosystem, where it is used to safeguard sensitive user data, such as encrypted backups [29].

AES can be used in various modes of operation, one such mode is Galois/Counter Mode (GCM) which uses universal hashing over a binary Galois field to provide authenticated encryption. The Counter Mode allows for parallelized implementations which makes it efficient and fast, and the Galois mode provides message authentication by generating an authorization tag (auth_tag) which verifies the authenticity of the data ensuring that the encrypted data has not been modified during storage or transmission [30].

The encryption process in AES-GCM involves several components. The plaintext is combined with a randomly generated 12-byte initialization vector (IV). This IV ensures that even if the same plaintext is encrypted multiple times, the resulting ciphertext will be different, thereby preventing pattern analysis. The encryption is performed using a secret AES key, which is generally 128, 192, or 256 bits long.

Along with the resulting ciphertext, the process also produces an auth_tag. For decryption, all components must be present [30].

As AES-GCM is efficient, secure and combines encryption with integrity verification in a single operation, it is highly suitable for applications requiring speed while maintaining high security standards.

5
Methodology

In this chapter, the implementation process for the product is presented. The project was divided into two parts; creating a Minimum Viable Product (MVP) based on existing modules, and improving the MVP towards a Final Product. Each part was examined in turn, together resulting in the final prototype of a web application with a FR-based authentication system.

As our thesis aims to develop a secure, user-friendly, and accurate application, we evaluated each of these objectives as summarized in Table 5.1. Detailed explanations of the evaluation methods are provided in the corresponding sections.

Table 5.1: Evaluation of the application's aims.

Aim	**Evaluation Method**	**Section**
Secure	Anti-Spoofing Tests	5.3.3
User-Friendly	User Tests	5.2.3
Accurate	FR Performance Tests	5.3

5.1 Minimum Viable Product

The MVP part of the project was carried out in order to establish a pipeline for the web application and test existing models for FD and FR. Development followed an iterative agile approach as it allowed for quick development and was conducted in three iterations; basic pipeline setup, FD integration, and the addition of FR with verification. The final version of the MVP consisted of a pipeline capable of storing user information in a database, using the chosen models for FD and FR, and presenting this functionality to the user in the form of a simple-to-use web application.

5.1.1 First Iteration

In the first iteration, we set up a pipeline with a simple web application that used a username- and password-based authentication approach. The application was developed using Flask, a widely adopted Python framework chosen for its ease of use and quick setup. For user authentication, backend logic was implemented to store user information in a relational SQL database hosted on Supabase, as it provided a

straightforward solution for concurrent storage. The frontend was built with HTML, JavaScript, and CSS, utilizing Bootstrap for styling.

5.1.2 Second Iteration

With the web application pipeline from the first iteration operational, the second iteration focused on integrating YuNet, the selected FD model. To enable real-time detection, a script was developed to activate the user's webcam and capture input images from the webcam frames. These images were decoded from their Base64 representation and converted into an OpenCV-compatible format. YuNet was then applied to analyze the images and detect facial bounding boxes. If faces were detected, rectangular markers were drawn on the image based on the bounding boxes, and the processed image was re-encoded as a Base64 string for further use.

Following this iteration, users could authenticate using a live webcam feed. However, it is important to note that no FR or verification methods were implemented at this stage, meaning the system could only detect the presence of a face but not verify its identity. The primary objective of this phase was to ensure that the model could reliably detect faces under varying conditions.

5.1.3 Third Iteration

At this stage, a web application pipeline was successfully implemented and integrated with FD. The third iteration focused on integrating FaceNet, the chosen model for FR. The model was implemented using the DeepFace library.

To prepare the face data captured by the FD system for FaceNet, pre-processing steps were introduced to enable feature extraction. After detecting faces, the image was processed and formatted to meet the input requirements of the FaceNet model. FaceNet then generated a feature vector, which was normalized using L2 normalization to ensure consistency in magnitude. This feature vector served as a compact representation of the face and was stored in the database alongside the user's credentials.

For verification, the similarity between stored and newly captured face vectors was evaluated, and authentication was granted if the match met a predefined threshold. By the end of this iteration, the system was able to identify and verify users based on their facial features, allowing them to register an account using their face and log in by verifying their identity through FR.

5.2 Final Product

The final iteration of the project introduced several major changes compared to the third iteration of the MVP. The first aspect was removing the password-based login system transitioning the system to a mandatory biometric authentication process.

Development then proceeded with the implementation of a new three step registration and improving account management, including the ability to update user information, delete account and an added timetable for monitoring account activity. To strengthen security, a One-Time Password (OTP) sign in option and end to end encryption were also added. Additionally, the FaceNet model was fine tuned using transfer learning to improve performance. These changes will be explained in greater detail below.

5.2.1 Account Registration and Management

To enhance user experience, the new three step registration process was implemented. This was done by separating the registration steps into different pages. The first screen is designated for entering the user's username and email address. The second screen is for capturing the facial image used for the biometric registration. The third and final screen presents a summary, allowing the user to confirm or go back to modify their credentials.

Users were given the ability to update their username and email from the dashboard, a feature to permanently delete their account as well as always being able to see their current username and email. Both update and delete were performed by using a unique user ID to locate the corresponding row in the database, the appropriate action was then executed using the Supabase Python Client Library to update the database.

To provide users with greater visibility into their account activity, a timetable was added to the dashboard. A new table in our database was created specifically to store these activity records. Each event includes a timestamp, a brief description of the action performed, what device type was used, and the unique user ID. This ID allows the system to retrieve and display all events associated with the account on the dashboard page.

5.2.2 Design of User Interface

One change made was to replace Bootstrap with Tailwind CSS. Tailwind's functional CSS approach allowed for an easier and more precise control over styling, enabling a custom look. While transitioning to Tailwind CSS we changed the design drastically, from a basic, visually plain design used in the MVP to a pastel gradient background with a modern tech aesthetic, with added logos and other design elements.

5.2.3 User Tests

To validate the intuitiveness and identify potential design flaws of the new interface, a series of user tests were carried out. A form was created to allow participants to book a time slot and was then shared with individuals from different departments on Chalmers who were personally known to the development team. This resulted in 11 individuals who signed up for the testing. Participants were faced with common use cases such as biometric account registration, login and account management, as well

as answering questions about the experience (see Appendix B). Feedback and observations collected from participants were discussed for further possible improvements.

By observing how users interacted with the system and gathering their feedback, several usability issues and areas for improvement were identified and a number of design adjustments were made. These included improvements in visual consistency across buttons and text elements, added and enhanced visual feedback and error messages, user readability, among others.

5.2.4 One-Time Password as an Alternative Sign-in Method

Since the extended product no longer supports password-based login, an alternative fallback method was implemented using one-time passwords (OTP) to handle situations where FR fails due to events outside the user's control (e.g., hardware failure). As the OTP can only be used once, it does not matter if a potential attacker has recorded the OTP and serves as the motivation of choice for an alternative authentication method for the application.

To implement the OTP mechanism, a Flask session was used to temporarily store a unique session identifier. After a predefined number of failed FR attempts, users are prompted to authenticate through OTP. They are asked to enter their email address, after which a 6-digit numerical OTP is randomly generated, encrypted, and stored in a temporary database table along with the session's identifier. The OTP is then sent to the user's email using an HTML-formatted message, composed with Python standard libraries MIMEText and MIMEMultipart. For delivering the email, a Python standard library called smtplib was used and configured with a sender email, sender password, smtp server, and port.

When the user enters the OTP received in their inbox, the application retrieves the encrypted OTP from the database, decrypts it, and verifies it against the user input, granting access if the values match.

5.2.5 Protecting Sensitive Information with Encryption

Since sensitive user information like email addresses and face features are stored in the database, building a secure application was important to guarantee user privacy and data integrity. To address this, the application was extended to support encryption of sensitive information. AES-GCM was chosen as the encryption method as it provides both authenticated encryption and the ability to check the integrity and authenticity of authenticated data. A custom implementation was developed using the Python cryptography package [31]. For each encryption operation, a random 12-byte initialization vector (IV) was generated. The plaintext data was then encrypted using a previously generated AES key, producing a ciphertext and an authentication tag (auth_tag). The ciphertext, IV, and auth_tag were together stored in the database to be retrieved when needed for decryption. The AES key however, was securely stored locally on the device running the web application, en-

suring that encrypted data in the database cannot be decrypted or manipulated without physical access to the device. See Figure 5.1 for an illustrative overview.

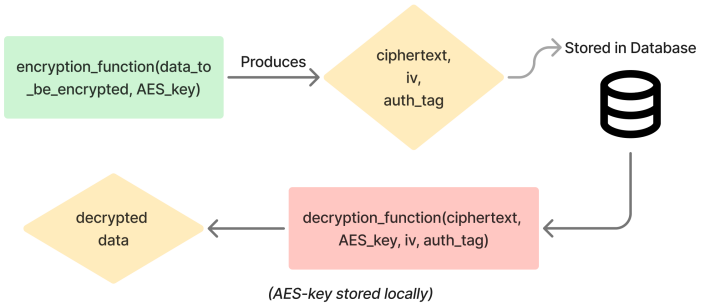

Figure 5.1: Overview of the application encryption process

The custom encryption implementation was integrated into three key areas of the application. First, during user registration, a Flask session temporarily stores information such as the email address, username, and a unique identifier. This session data persists only during the registration process and was automatically cleared after five minutes of inactivity or upon leaving the registration page. However, due to the large size of the extracted facial features captured during the detection phase, this data cannot be stored directly in the session. Instead, it was encrypted and stored in a temporary database table. The unique identifier in the session was later used to retrieve and associate encrypted facial data in the final step of account creation. Secondly, the encryption mechanism was applied during account creation, where all sensitive user data was encrypted before being stored in the main user table of the database. Lastly, encryption was also used in the OTP process presented in Section 5.2.4. The OTP was encrypted and stored in a temporary database table, similar to the one used for facial feature data.

5.2.6 Improving Models with Transfer Learning

To try to enhance the performance of the FR pipeline, specifically for real-world webcam images even further, transfer learning was applied to the standard 512-dimensional FaceNet model using the Georgia Tech Face Database, which is reviewed in Section 5.3.2. This section outlines the approach taken as an attempt to fine-tune the model for webcam images in particular.

The fine-tuning process began by initializing the FaceNet model with its original weights. To retain the general features learned during initial training, the majority of the model layers were kept frozen, while only the final 30 of 447 layers were set to be trainable. Additionally, all Batch Normalization layers were frozen in order

to avoid instability during fine-tuning, caused by small batch sizes because of the small dataset. This strategy aimed to ensure that the model could adapt to the new dataset consisting of webcam images, while maintaining robust and generalizable features.

Fine-tuning was carried out using an optimizer with a learning rate of 10^{-6}. A low learning rate was selected to allow gradual and stable updates to the weights in order to try to minimize the risk of overfitting or catastrophic forgetting. During training, the same triplet loss function as FaceNet was used, with the objective of drawing feature vectors of images of the same identity closer while feature vectors from different identities are pushed apart. To promote a balance between distinctiveness and a tolerance for minor variations in the face images, a margin of 0.1 was chosen for the triplet loss, meaning the distance between the feature vectors from the images of the same identity has to be lower than the distance between the feature vectors from the different identities with a value of at least 0.1 to have no loss. Training was performed for only one epoch to prevent overfitting on the small dataset, comprising of 100 steps and a batch size of 32.

An important improvement to the training stability and effectiveness was the use of semi-hard triplet selection. Instead of selecting triplets randomly, semi-hard triplets were dynamically generated by selecting negatives that were harder than easy negatives but still more distant than positives by a margin. Feature vectors for all face images were computed in advance, and triplet selection was based on their vector distances to satisfy the semi-hard condition. This method ensured that training focused solely on challenging examples that are neither too easy or too difficult, attempting to decrease the risk of overfitting and improving the convergence speed and final model quality.

The Georgia Tech dataset was used for fine-tuning and model evaluation. Preprocessing of the images from the dataset before passing them through the model for training was done similarly to how it is done in the actual FR pipeline of the system to avoid inconsistencies. To see how the model performance changed during training, the model was evaluated on the dataset again after fine-tuning to see how the current training setup affects the model. The evaluation after the fine-tuning was an important step to help assess how the training structure could be improved to produce a fine-tuned model of higher quality.

In a similar manner to the fine-tuning approach applied to the FaceNet model, an attempt was made to improve the performance of the anti-spoofing model through additional training. For this purpose, the MSU-MFSD [32] dataset was extracted and intended to be used for training. However, due to practical challenges during implementation, this fine-tuning process was not successfully completed within the project timeline. As a result, no performance evaluation or conclusive analysis could be carried out for this attempt.

5.3 Performance Testing

A high-performing FR system is one that accurately verifies users, which requires identifying optimal settings. Since FR system performance typically varies with the verification decision threshold, extensive testing was conducted to evaluate the model across different thresholds and determine the most effective values and their corresponding performance.

Interpreting individual test case outcomes was straightforward. If the system predicted that two face images belong to the same person, this was considered a positive, and if it predicted they belong to different people, this was considered a negative. Testing the FR functionality on an image dataset yielded predictions where each prediction was classified as a true positive (TP), true negative (TN), false positive (FP), or false negative (FN).

- **TP**: The system correctly verified that two images belong to the same person.
- **TN**: The system correctly rejected a match when two images belong to different people.
- **FP**: The system incorrectly verified two different persons as the same person.
- **FN**: The system incorrectly rejected a match between two images of the same person.

Since FNs prevent legitimate users from authenticating, while FPs pose a security risk by granting unauthorized access, system performance depends on minimizing both. Therefore, tracking these errors across a range of verification thresholds provided valuable insights and helped determine whether threshold adjustments were necessary.

5.3.1 Testing Procedure

To evaluate how well the application performs, the FR pipeline must be tested on entire datasets of face images. Since the system uses verification for the inference step, it predicts one-to-one matches to authenticate users. Each test case thus consisted of a pair of feature vectors extracted from the dataset using our own module. To ensure the system was tested across all possible outcomes (TP, FP, FN and TN), the test cases consisted of feature vector pairs either from the same person (evaluating TP and FN) or from different persons (evaluating TN and FP). The expected result was positive if the pair came from the same person and negative otherwise.

Since the most common use case for the face recognition (FR) functionality is assumed to involve comparing two images of the same person, such test cases constituted 80% of the total. Although comparisons between images of different individuals likely represent less than a fifth of real-world authentication attempts, maintaining a certain level of class distribution balance is important for obtaining reliable accuracy scores. In highly imbalanced datasets, a naive classifier, such as a dummy classifier that always predicts the majority class, can achieve deceptively

high accuracy without truly reflecting the model's performance.

Running all test cases through the verification module produced a list of predicted values, which were then compared to the expected values to determine the outcome of each prediction. This data allowed for visualizing the results using confusion matrices and computing the following common performance metrics (see Table 5.2): accuracy, precision, recall, and F1-score. To achieve a suitable balance between security and usability, multiple tests were conducted at different decision thresholds, analyzed using a receiver operating characteristic (ROC) curve.

Table 5.2: The common evaluation metrics.

Metric	**Description**	**Formula**
Accuracy	The ratio of correct predictions (both TPs and TNs) to the total number of predictions.	$\frac{TP+TN}{TP+TN+FP+FN}$
Precision	The ratio of TPs to the total number of predicted positives. Measures how many of the predicted positive matches are correct.	$\frac{TP}{TP+FP}$
Recall	The ratio of TPs to the total number of actual positives. Shows how many of the actual positive matches were correctly identified.	$\frac{TP}{TP+FN}$
F1-Score	The harmonic mean of precision and recall, providing a balance between the two.	$2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision}+\text{Recall}}$

These metrics are essential for evaluating system performance in different ways: accuracy provides an overall measure of correctness, precision and recall offer insights into the application's ability to avoid FPs and FNs, respectively, while the F1-score measures whether both high recall and high precision are achieved. The idea was to observe how these metrics changed across the different thresholds in order to find the settings that yield optimal face verification performance.

A confusion matrix is a table that displays the counts of TPs, FPs, TNs and FNs (see Figure 5.2). It provides a clear overview of how many predictions were correct and incorrect. From this matrix, the performance metrics mentioned above can be derived, allowing for a deeper understanding of the verification module's behavior.

The ROC curve visualizes the trade-off between the true positive rate (TPR) and the false positive rate (FPR) across different decision thresholds. In the context of face verification, it illustrates how well the system distinguishes between genuine users and impostors. A steep curve toward the top-left corner indicates strong performance, with high TPR and low FPR. This helps in selecting an optimal threshold that balances security (minimizing FPs) and usability (minimizing FNs).

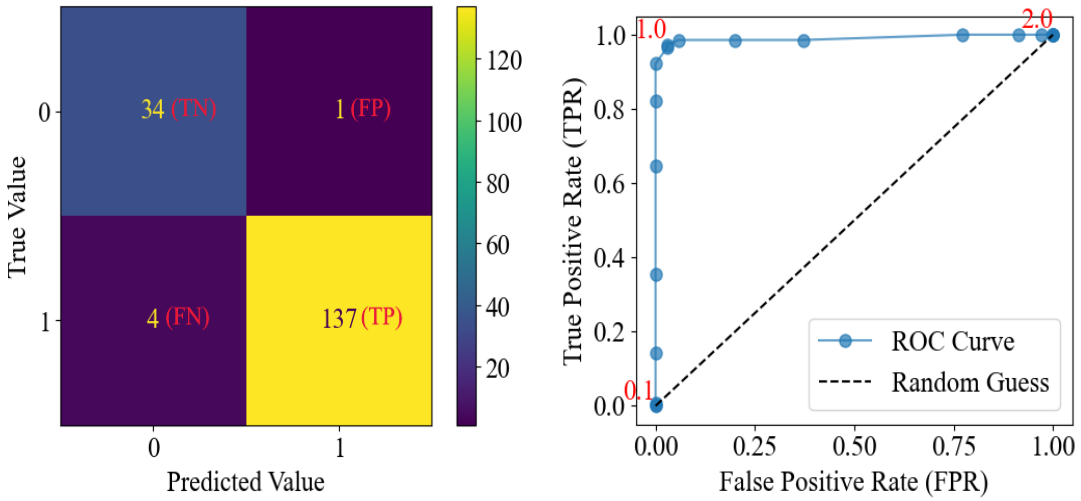

Figure 5.2: Examples of a confusion matrix and an ROC curve.

Additionally, the average time it took for the application to detect a face and extract its features (including potential alignment and anti-spoofing detection) was measured, along with the frequency of undetected and misaligned faces. These metrics capture the application's usability by assessing how quickly it processes faces, while also providing insight into how often it fails before even reaching the critical verification step.

All tests and experiments were conducted on a laptop running Windows 10 Home (64-bit), equipped with an AMD Ryzen 5 5500U CPU, 8 GB of RAM, and no dedicated GPU.

5.3.2 Image Datasets

Since the most common real-world input is expected to be webcam face images or smartphone selfies, it was considered ideal to test the application's performance on datasets consisting solely of such images. However, testing on the commonly used, celebrity-laden LFW dataset still had merit, as it provided a valuable performance baseline. Given that the performance of the employed model is already well-documented on LFW, testing on it allowed us to verify that our implementation behaved as expected before moving on to more application-relevant datasets.

Two such datasets were used for testing: the Georgia Tech Face Database from the Center for Signal and Image Processing at Georgia Institute of Technology [33] and the FEI Face Database from the Artificial Intelligence Laboratory at FEI in São Bernardo do Campo, São Paulo, Brazil [34]. The datasets contain 750 and 2,800 images, respectively, and both include a subset of images per person that resemble webcam-quality. Each subset includes one reference image captured under good lighting and with a proper face and body angle, along with several images of gradually decreasing quality – such as the person turning their face away or being

in poorer lighting conditions. Test cases involving the same person were created by pairing the reference image with each of the other images in the same subset, while test cases involving different persons were formed by randomly selecting non-identical pairs of reference images.

Regarding diversity, the FEI dataset has an equal split between male and female subjects and includes individuals from various racial backgrounds. The Georgia Tech dataset contains fewer women but remains racially diverse. However, both datasets include relatively few individuals of African descent, which may limit the generalizability of the performance results across all demographics. This matters because the goal of developing a high-performing system should arguably include ensuring reliable performance across diverse user groups. A lack of representation can lead to biased outcomes, reducing the system's effectiveness for underrepresented populations.

5.3.3 Anti-Spoofing Testing

Since the Final Product features face anti-spoofing (FAS) capabilities in the form of an additional model within the FR pipeline, finding and evaluating its most effective decision thresholds was essential. With a test dataset consisting of both real and spoofed images, the procedure was straightforward: the FD module extracted the face from each image, after which the FAS module assessed the legitimacy of the input across a range of decision thresholds.

The classification of each prediction followed the same logic as in the verification tests, with slight contextual adjustments:

- **TP**: A real image has been correctly classified as real.
- **TN**: A spoofed image has been correctly classified as spoofed, preventing a spoofing attack.
- **FP**: A spoofed image has been incorrectly classified as real, allowing a potential attack to proceed.
- **FN**: A real image has been incorrectly classified as spoofed, possibly denying access to a genuine user.

Just like in FR, both FPs and FNs are undesirable outcomes in anti-spoofing systems, as neither granting attackers access nor denying legitimate users should occur. A predominance of either type of error suggests the model may require further tuning. High FPs indicate the model is too lenient, while high FNs suggest it is too strict.

During the test, once a list of predictions had been compiled, it was compared to the expected values for each image. Using this data, the same metrics as before (accuracy, precision, recall, and F1-score) were calculated, along with an ROC curve and confusion matrices to provide a clearer view of the prediction distribution.

The data used for these tests came from the Large Crowdcollected Facial Anti-Spoofing Dataset (LCC FASD), which consists of 18,827 images, mostly of webcam or smartphone quality: 16,885 depicting real faces and 1,942 depicting spoof attempts [35]. The spoofed images are mostly smartphone photos of computer screens displaying face images. The data is divided into development, training, and evaluation sets. Accordingly, the evaluation data was used for testing the FAS model. It features a diverse range of individuals, although older people are somewhat underrepresented. The evaluation set contains 314 real and 7,266 spoofed images. Since increasing the threshold in the FAS model makes it more difficult to classify an image as real, this highly imbalanced distribution could have yielded misleading results. A trivial predictor could, for instance, have achieved around 95% accuracy by always predicting the majority class. To address this, a randomized sample of 700 spoofed images (without replacement) was selected for testing, alongside all 314 real images, to create a more balanced class distribution. In total, 1,014 images were tested each run. To ensure the results were not overly dependent on a single random sample, the test was repeated multiple times, allowing the model to be evaluated based on a more representative average across the many available spoofed images.

6
Results

This chapter presents the key outcomes of the development and evaluation of our web application. The results are organized according to the two main development stages: the Minimum Viable Product (MVP) and the Final Product. For each stage, we report both technical performance metrics and qualitative aspects related to user experience and system usability. Emphasis is placed on the performance of the FR and anti-spoofing components, evaluated across multiple datasets.

6.1 Minimum Viable Product

The MVP phase resulted in a fully functioning web application prototype that enabled user authentication via either password or FR. This section first describes the user interface and the intended user flow, outlining how users interact with the system. Following that, the performance of the integrated pre-trained models for FD and FR is presented, based on evaluation across three different face datasets.

6.1.1 User Interface and User Experience

The web application consists of six main pages: a homepage, two separate registration pages (one for password-based registration and one for FR), two corresponding login pages and a user dashboard.

On the homepage, users are presented with options to either log in or register using a standard password-based approach or with FR, as shown in figure 6.1. Depending on the chosen method, the user is directed to the corresponding registration page. In the password-based registration, users provide an email address and a password. In the FR registration, users only provide an email address and then have their facial data captured for future verification.

Similarly, there are two separate login pages—one for each authentication method. For FR login, the user is prompted to enter their email address, after which the system activates the webcam, captures and processes an image, and verifies the user's identity, as shown in figure 6.2. Upon successful authentication through either method, the user is redirected to a dashboard page where account information is displayed and a logout option is available.

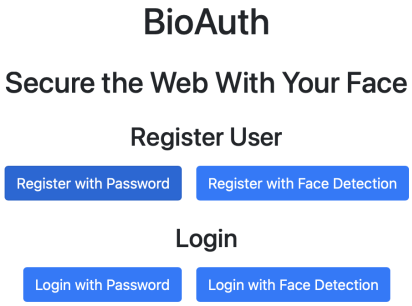

Figure 6.1: The homepage for the MVP

The interface of the web application is minimalistic and implemented using Bootstrap. It is designed to be visually clear and easy to navigate. At this stage, no user testing has been conducted, which indicates that the design has some areas for improvement for the Final Product.

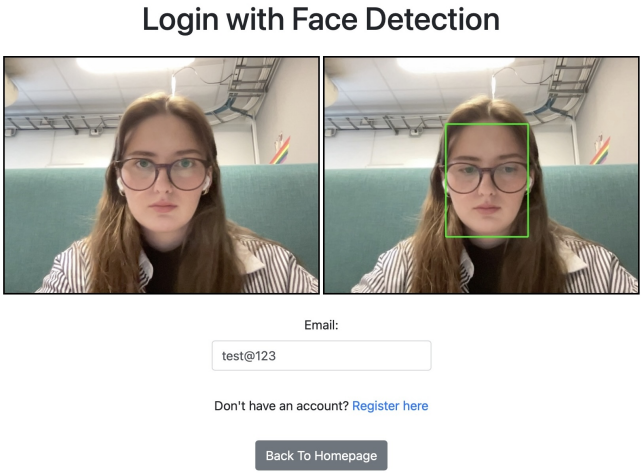

Figure 6.2: The FR login page for the MVP

6.1.2 Performance of the Face Recognition Pipeline

Below are the performance results from evaluating the MVP on the LFW, FEI, and Georgia Tech face datasets. The threshold range was defined from 0 to 2.0, with increments of 0.1.

Running the LFW dataset through the test module yielded 9,215 test cases. Evaluating these test cases produced the ROC curve shown in Figure 6.3.

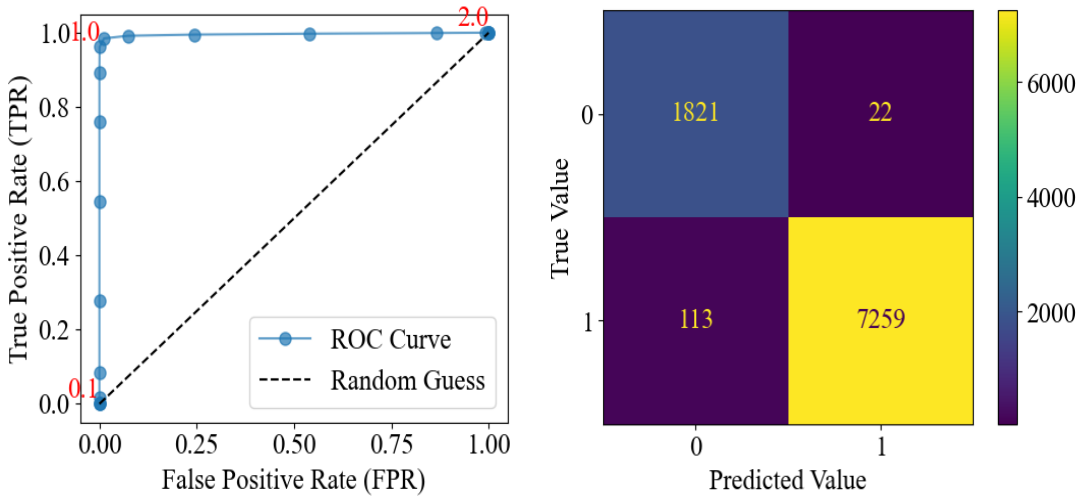

Figure 6.3: ROC curve and confusion matrix for a threshold of 1.1, evaluated on the LFW dataset.

According to the curve, the optimal thresholds lie approximately between 0.8 and 1.3. These thresholds correspond to the performance metrics presented in Table 6.1.

Table 6.1: Performance metrics for the system evaluated on the LFW dataset.

Threshold	Accuracy	Precision	Recall	F1-Score
0.8	0.8091	1.0000	0.7614	0.8645
0.9	0.9150	1.0000	0.8938	0.9439
1.0	0.9691	0.9999	0.9615	0.9803
1.1	0.9853	0.9970	0.9847	0.9908
1.2	0.9787	0.9819	0.9917	0.9868
1.3	0.9473	0.9423	0.9950	0.9679

The highest scores across all metrics are observed in the middle of the threshold interval. This is exemplified by the confusion matrix in Figure 6.3 for a threshold value of 1.1, which shows 135 (1.47% of the total amount) incorrect predictions. Furthermore, precision decreases as the threshold increases, while recall increases. The F1-scores follow a similar trend to accuracy, with the highest values occurring at the mid-range thresholds.

Table 6.2: Performance metrics for the system evaluated on the FEI dataset.

Threshold	Accuracy	Precision	Recall	F1-Score
0.7	0.8700	1.0000	0.8376	0.9116
0.8	0.9356	1.0000	0.9196	0.9581
0.9	0.9678	0.9996	0.9602	0.9795
1.0	0.9778	0.9972	0.9750	0.9860
1.1	0.9747	0.9882	0.9801	0.9841
1.2	0.9575	0.9644	0.9832	0.9737

For the FEI dataset, which generated 3,201 test cases, similar patterns can be observed in the performance metrics shown in Table 6.2. The ROC curve in Figure 6.4 suggests that optimal threshold values lie between 0.7 and 1.2. A threshold of 1.0 appears to achieve relatively high performance metrics, and thus produce small numbers of both FPs and FNs, as shown in the confusion matrix in Figure 6.4.

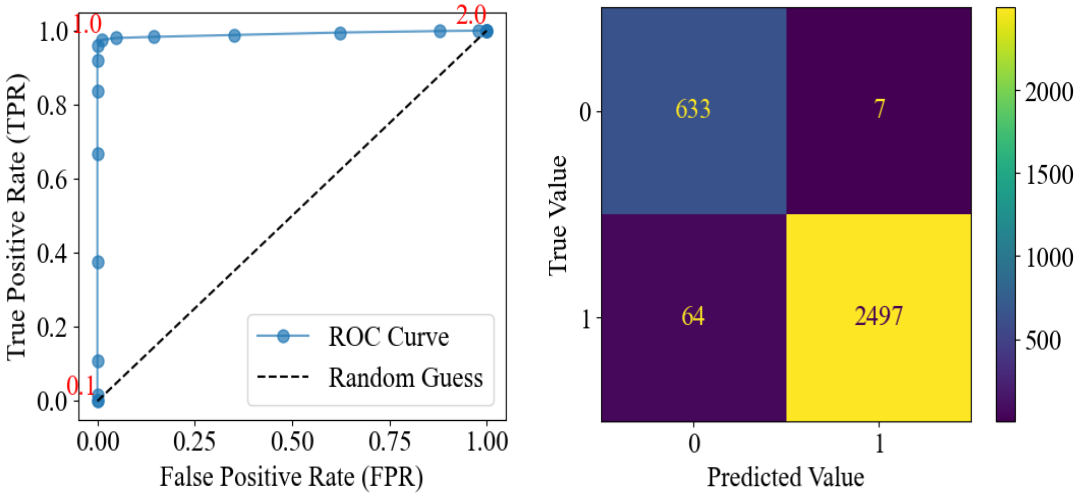

Figure 6.4: ROC curve and confusion matrix for a threshold of 1.0, evaluated on the FEI dataset.

Table 6.3 shows that the MVP achieves its highest performance scores on the 875 test cases from the Georgia Tech dataset. Not only does a particular threshold yield the highest value for each metric, but near-optimal performance is sustained across the threshold range 0.9 to 1.1. At thresholds 0.9 and 1.0, both precision and recall are either perfect or within a few ten-thousandths of perfection.

Table 6.3: Performance metrics for the evaluation on the Georgia Tech dataset.

Threshold	Accuracy	Precision	Recall	F1-Score
0.7	0.9143	1.0000	0.8929	0.9434
0.8	0.9771	1.0000	0.9714	0.9855
0.9	0.9977	1.0000	0.9971	0.9986
1.0	0.9977	0.9972	1.0000	0.9986
1.1	0.9920	0.9901	1.0000	0.9950
1.2	0.9760	0.9709	1.0000	0.9852

Although the metrics are either higher than or comparable to the highest metrics observed on the other two datasets, the typical pattern of increasing FPs and decreasing FNs with higher threshold values remains evident. Moreover, Figure 6.5 shows that the threshold yielding the highest average performance scores still results in two false positives.

The system's responsiveness and robustness during the face detection and feature extraction steps were also evaluated. Table 6.4 presents the average time required

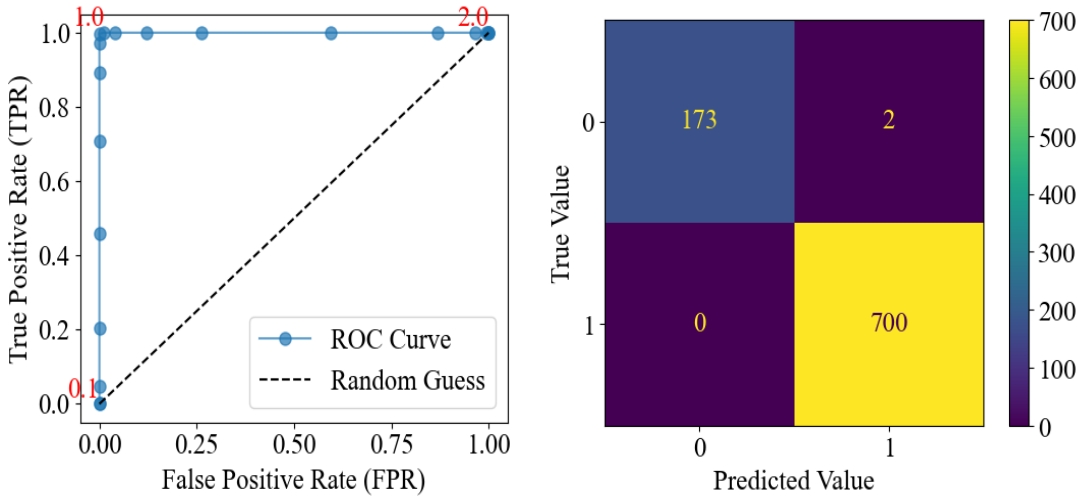

Figure 6.5: ROC curve and confusion matrix for a threshold of 1.0, evaluated on the Georgia Tech dataset.

for these operations per image, along with the number of undetected and misaligned faces for each dataset.

Table 6.4: Face detection and feature extraction performance.

Dataset	Average Time per Image (s)	Undetected Faces	Misaligned Faces	Total Images
LFW	0.1323	132	3	13,233
FEI	0.1514	38	0	2,800
Georgia Tech	0.1532	0	0	750

6.2 Final Product

After completing the MVP, multiple changes were made, together resulting in the Final Product. The main focus for the changes were design, adding an anti-spoofing model and other features to improve the security of the authentication system. This section describes the results of these changes in more detail.

6.2.1 User Interface and User Experience

Between the MVP and the Final Product a lot of changes were made on the design to improve the user experience. The final version of the web application consists of four main pages; a homepage, a registering page divided into different steps to guide the user, a login page and a user dashboard. If a HTTP 404 error occurs, where a file or page isn't found, a custom error page has also been created. This provides the user with sufficient information to get back to a working page along with being more aesthetically pleasing than the default 404 page.

The overall feedback from the user tests suggested that the system worked as intended in most areas. Users sometimes experienced uncertainty around certain interactions and some design elements were a bit inconsistent. These findings allowed us to improve the overall experience for our future users. With this feedback in mind, further improvements of the UI was made focusing on placement, sizing and hovering actions for the buttons in the application. These adjustments were made to make the application more intuitive and consistent throughout the different pages. The final version is visualized and its functionalities are explained below.

On the homepage, shown in Figure 6.6, users are presented with options to register an account or to log in using a previously registered account. The homepage also contains an informative section which describes the project and links to our open-source Github repository.

Figure 6.6: The homepage for the Final Product

When registering an account, the user is guided through three different steps shown in Figure 6.7, 6.8 and 6.9. The first step is providing a username and an email address. The next step is to take an image of their face following the instructions presented to make sure the image is of good quality in terms of placement and lighting. The user can retake the image if they want to before finalizing the registering. In the last step, the credentials and the taken image are shown so that the user can make sure everything is correct before creating the account.

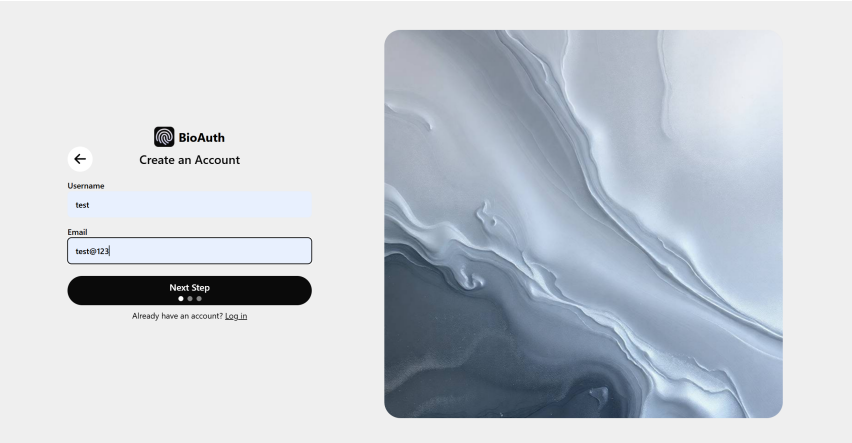

Figure 6.7: The first step for registering an account in the Final Product

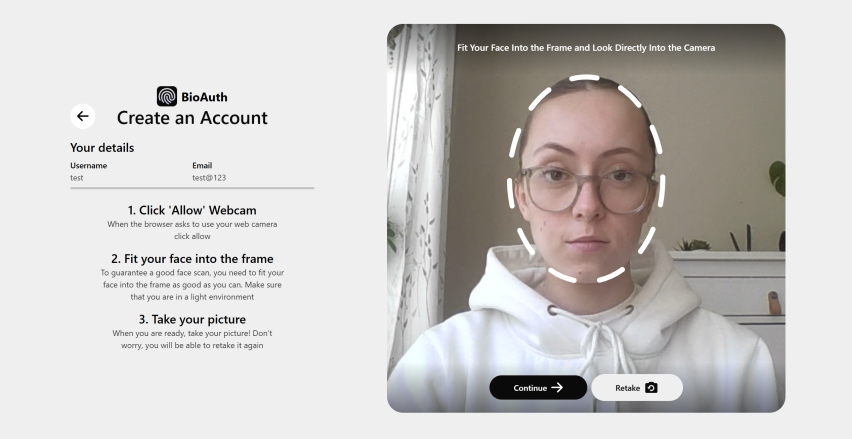

Figure 6.8: The second step for registering an account in the Final Product

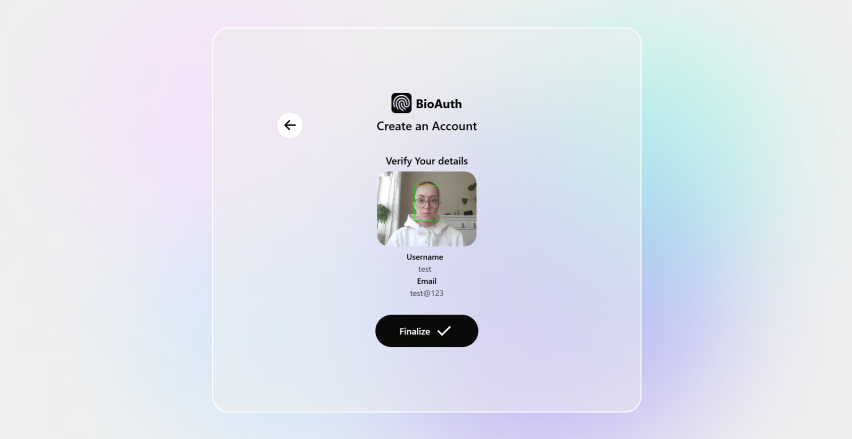

Figure 6.9: The third step for registering an account in the Final Product

The login page is similar to the first registration page. Here, the user can choose between using their username or email to log in. After providing a registered username or email, the image data from the webcam is collected and shown in the image

field. Once a match is verified, the user is directed to the user dashboard.

The user dashboard is shown in Figure 6.10 and consists of account details where the username and email is shown. Next to them a button for account settings is placed and when pushed, the popup window shown in Figure 6.11 appears. This includes possibilities to update the username and email along with the option to delete the account. A scrollable timelog is also placed on the dashboard, where the user can see the activity for the account. The events that are shown is when the account is registered, login occurrences and updating of the username or email. The events are followed by a timestamp and the type of device that was used for each of them. The last option the user is provided with on the dashboard is to click the logout button at the bottom of the page which redirects them to the homepage again.

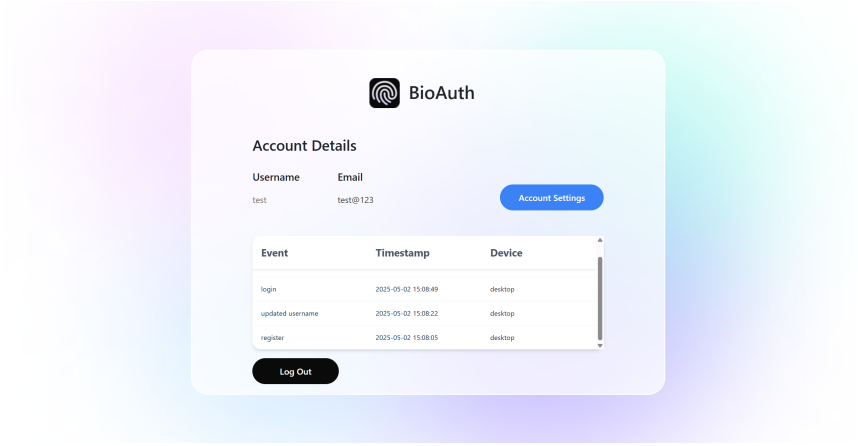

Figure 6.10: The account dashboard for the Final Product

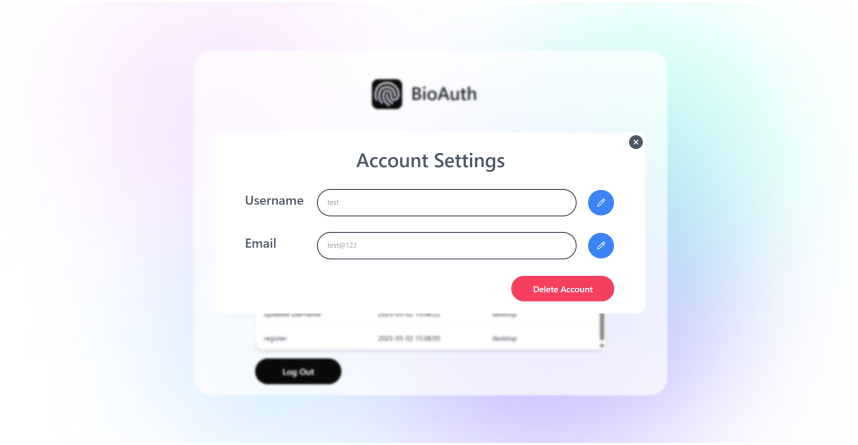

Figure 6.11: The account settings for the Final Product

6.2.2 Security Features

Several efforts were made towards security features and the final application included measures to counter spoofing attacks, sensitive data exposure, and SQL injections,

as well as a secure secondary authentication method through One-time passwords. An anti-spoofing model was implemented to detect false substitutes of an authorized person's face and block unauthorized login attempts. To prevent exposure of sensitive data, AES-GCM encryption was added throughout all processes in the application where sensitive data was at risk. For SQL injections, Supabase's query builder provided the necessary query sanitization as it internally relies on parameterized queries to safely handle user input and prevent the execution of malicious SQL. Additionally, client-side error messages were sanitized and standardized to predefined responses. This was done to avoid exposing any internal application logic, thereby reducing the risk of potential attackers gaining insight into how the system operates.

6.2.3 Performance of the Tuned Face Recognition Pipeline

The Final Product includes the option to use fine-tuned weights obtained through transfer learning on the original FaceNet FR model. It also introduces stricter alignment detection than the MVP to ensure that only images depicting well-positioned faces are used for feature extraction. Since preliminary testing revealed that a significant portion of the test dataset contained poorly aligned images, performance was evaluated both with and without alignment detection enabled to provide a more comprehensive assessment.

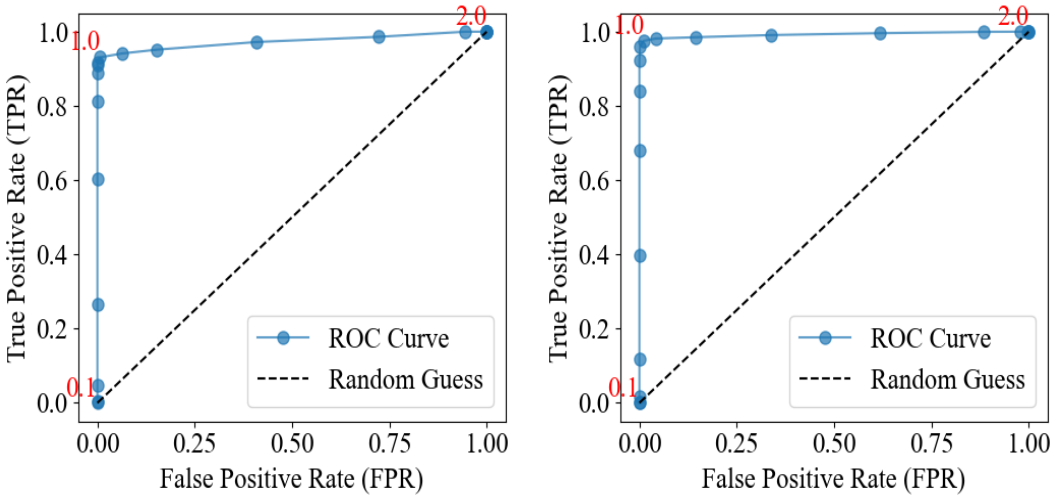

Figure 6.12: ROC curves for the test on the FEI dataset with (left curve) and without (right curve) alignment detection.

When tested on the FEI dataset, the performance metrics progression across thresholds, both with and without alignment checks, show little deviation from that observed for the MVP. This is illustrated in Tables 6.5 and 6.6, and in the ROC curves shown in Figure 6.12.

Notably, the accuracy for the subset of images that passed the alignment check is consistently a few percentage points lower than for the full, unchecked image set. Examples of confusion matrices and results from testing on the LFW dataset, which

Table 6.5: Performance metrics for the FEI test with alignment detection.

Threshold	Accuracy	Precision	Recall	F1-Score
0.7	0.9101	1.0000	0.8877	0.9405
0.8	0.9281	1.0000	0.9102	0.9530
0.9	0.9336	1.0000	0.9171	0.9568
1.0	0.9433	0.9981	0.9309	0.9634
1.1	0.9405	0.9838	0.9413	0.9620
1.2	0.9308	0.9616	0.9516	0.9566

Table 6.6: Performance metrics for the FEI test without alignment detection.

Threshold	Accuracy	Precision	Recall	F1-Score
0.7	0.8729	1.0000	0.8411	0.9137
0.8	0.9375	1.0000	0.9219	0.9594
0.9	0.9681	0.9996	0.9606	0.9797
1.0	0.9781	0.9972	0.9754	0.9862
1.1	0.9769	0.9894	0.9816	0.9855
1.2	0.9591	0.9644	0.9852	0.9747

closely resemble those of the MVP, are included in Appendix A. The Georgia Tech dataset was excluded from performance evaluation, as it was used for transfer learning, meaning that testing on it would not measure the model's ability to generalize to unseen data.

Table 6.7: Face detection and feature extraction statistics with the Final Product alignment detection.

Dataset	Average Time per Image (s)	Undetected Faces	Misaligned Faces	Total Images
LFW	0.1489	132	6,197	13,233
FEI	0.1681	39	1,917	2,800

As shown in Table 6.7, the updated alignment detection excludes 46.8% of the LFW images and 64.5% of the FEI images from feature extraction, thereby reducing the total number of test cases.

6.2.4 Anti-Spoofing Testing

Testing the untuned FAS model on the LCC FASD dataset, using randomly sampled spoofed images, was repeated ten times. This time, the threshold range was set between 0.90 and 1.00, with incremental steps of 0.01.

As indicated by the ROC curve in Figure 6.13, a relatively strong balance between TPR and FPR appears to cluster in the narrower threshold range of 0.92 to 0.98. The average scores for those thresholds are presented in Table 6.8. Accuracy remains

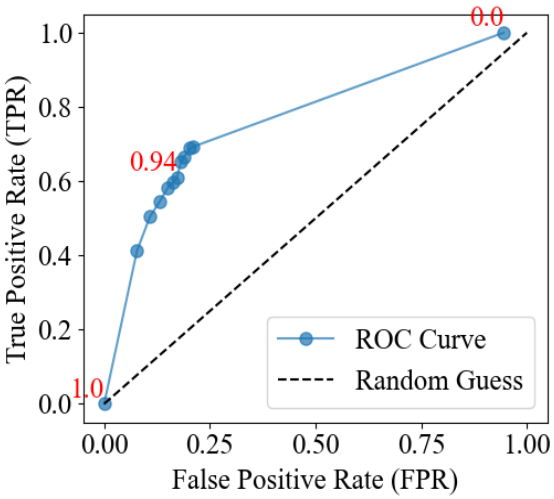

Figure 6.13: Average ROC curve for the FAS tests on the LCC FASD dataset.

fairly stable across that range at around 76%, but the threshold progression also highlight increasing precision and decreasing recall, similar to what was observed in the MVP performance test results.

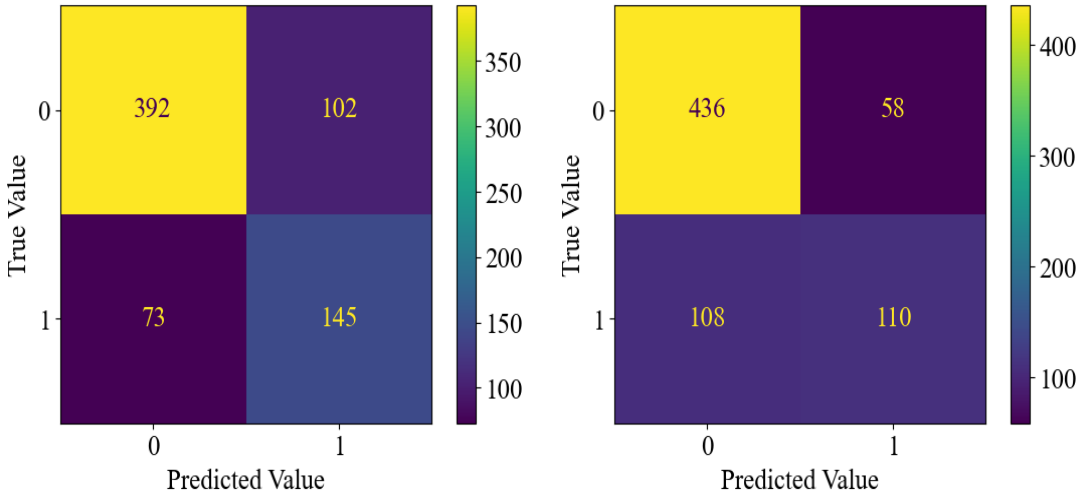

Figure 6.14: Confusion matrices for thresholds of 0.92 (left) and 0.98 (right).

This precision-recall trade-off is illustrated by the confusion matrices in Figure 6.14, which come from the same test run. The matrix on the left represents a threshold of 0.92, and the right one, 0.98. Comparing the two shows how the number of FNs and FPs almost switches between the different thresholds.

Furthermore, on average, face detection failed for 96 images from the real subset of the test data and 200 images from the spoofed subset. This corresponds to 30.57% of the real-face images and 28.57% of the spoofed images being excluded from the

FAS testing. As a result, an average total of 718 images remained for testing.

Table 6.8: FAS average accuracy metrics for the LCC FASD data across the selected thresholds.

Threshold	Accuracy	Precision	Recall	F1-Score
0.92	0.7667	0.6057	0.6651	0.6340
0.93	0.7677	0.6103	0.6514	0.6301
0.94	0.7617	0.6075	0.6101	0.6087
0.95	0.7642	0.6158	0.5963	0.6058
0.96	0.7684	0.6284	0.5826	0.6045
0.97	0.7702	0.6438	0.5459	0.5907
0.98	0.7741	0.6708	0.5046	0.5758

7
Discussion

This chapter reflects on the outcomes, challenges, and design decisions made throughout the development of the FR system. By analyzing the system's performance results, limitations, ethical considerations, and areas for future development, we aim to provide a balanced perspective on its practical viability. The discussion highlights both the strengths of the final product and the areas that would benefit from further refinement or research. This analysis also relates to each aim and the outcome of its corresponding evaluation, as outlined in Table 5.1.

7.1 System Performance

Looking at the performance of the implemented FaceNet-based face recognition system, the best accuracies for the FEI (97.81%), Georgia Tech (99.77%), and LFW (98.92%) datasets are all close to or above both the human-level accuracy (97.53%) and the official FaceNet benchmark accuracy (99.63% on LFW), as referenced in Section 1.1 and Chapter 3. The slight discrepancy between our LFW result and the benchmark is likely due to differences in test case generation. Notably, the results from the FEI and Georgia Tech datasets are the most indicative of high real-world FR performance, as the image conditions in these datasets closely resemble those expected as input in our application.

The Final Product introduced both fine-tuned weights and stricter alignment detection. The effect of the fine-tuned weights appears to be negligible, as a comparison between the results of the Final Product without alignment detection and the MVP shows that the FEI and LFW accuracies changed by only +0.03% and -0.09%, respectively. The addition of alignment detection seems to have had a mixed impact. While it increased the best LFW accuracy by 0.39%, it also led to a noticeable drop in FEI performance in general. This is somewhat surprising, as a better-aligned subset, where the full face is visible, would be expected to improve feature extraction. However, since the alignment detection excluded 64.5% of the FEI images, the reduced dataset size may have limited the reliability of the performance metrics.

Moreover, all top accuracies, and indeed the best average performance metrics, occur at verification thresholds of 1.0 or 1.1. However, a clear trade-off between precision and recall is evident across all performance tests for both the MVP and the Final Product. While both metrics remain relatively high around a threshold of 1.0, neither is perfect. This highlights the need to balance the risk of false FPs and

FNs - in other words, the choice between mistakenly granting access to illegitimate users or denying access to legitimate ones. A lower threshold can eliminate the risk of authenticating illegitimate users (resulting in perfect precision) but increases the likelihood of rejecting legitimate users (lower recall), whereas a higher threshold does the opposite.

It may seem intuitive to prioritize security above all and reject any possibility of false positives. However, if legitimate users repeatedly fail to authenticate, the system may quickly lose their trust and perceived reliability. This is why perfect precision must be accompanied by reasonably high recall to ensure the system is both high-performing in terms of accuracy and user-friendly.

Also, there does not appear to be a perfect threshold. Even in the case of the Georgia Tech dataset, which yielded the highest overall performance metrics, Figure 6.5, showing the best-performing confusion matrix, underscores a residual risk of mistakenly granting access to an unauthorized user. This holds true even when working with what seems to be a near-ideal dataset of face images.

The anti-spoofing test results showcased a similar trade-off between precision and recall to that for FR. In the case of the FAS model, the tuning decision between allowing more spoof attacks to pass (more FPs) or rejecting more genuine authentication attempts (more FNs) is highlighted. The key difference from the FR performance results is the scale: the FAS model results in a greater proportion of both false negatives and false positives, regardless of which type is minimized through threshold adjustment. However, as the F1-score tends to drop with higher threshold values, minimizing the total number of FNs and FPs appears to correspond with selecting a lower threshold within this range.

With an accuracy of roughly 76% and precision and recall values around 60%, using the FAS model as part of the deployed application would be problematic. While it is capable of detecting some spoofing attempts, it remains relatively prone to both rejecting genuine users and accepting spoofed inputs. This trade-off reduces its practical usability: although it adds a layer of security, the frequency of false rejections significantly undermines the user experience. The difference in accuracy compared to the FAS model's advertised benchmark of 93% [26] is substantial and indicates that the model is not suited for webcam-quality images.

7.2 Limitations

While the system meets its primary functional goals, several aspects of the development and evaluation process presented limitations that are important to acknowledge. This section outlines areas where improvements could be made, including the choice of models, scope and methodology of user testing, and the approach to fine-tuning the feature extraction model. These reflections help clarify the boundaries of the current implementation and point to potential directions for further refinement.

7.2.1 Choice of Models

Due to the vast number of existing models for FD, FR and anti-spoofing, it is possible that some relevant approaches were overlooked during the research phase. This raises the possibility that a fully comprehensive comparison was not achieved, despite efforts to include a wide range of critically reviewed sources. A more extensive review of the literature might have identified alternative models, potentially influencing the system's performance. Furthermore, this project utilized open-source, pre-trained FR models, due to considerations of cost and accessibility. While these models delivered satisfactory results, proprietary models may offer advantages in terms of accuracy, robustness, or mitigation of demographic biases. With the possibility of investigating commercial models an evaluation of the trade-offs between open-source and commercial FR solutions could be made. For areas such as security, efficiency, and fairness across different user groups this would make an interesting addition to the overall evaluation. Despite these limitations, the selected models and performance testing results are considered sufficient to meet the objectives of this project.

7.2.2 User Tests

We carried out the user test with students from Chalmers University of Technology, from a range of different academic educations and ages between 21 and 28. Although the participants were from different backgrounds, as students at Chalmers and their age suggested them to have a baseline of experience with technology. While this was useful for understanding how people comfortable with technology interacted with the system, it may not fully reflect the experience of individuals with less technical background. As a result, some issues related to understandability and onboarding may have been underrepresented.

Another limitation was the number of participants, since the group tested on was relatively small, this could limit the generalizability of the feedback. Additionally, all participants had a personal connection with at least one member of the research team. This could result in their feedback being overly positive, and not represent their thoughts accurately.

The user tests were made up of clear, predefined tasks that offered the participants a straightforward way to interact with the application. This consistency was beneficial because it ensured that all participants had received the same instructions. However, it may have created an idealized scenario, potentially providing a misleading impression of the usability and overlooked underlying issues.

7.2.3 Challenges in Fine-Tuning FaceNet with Limited Data

The goal of fine-tuning FaceNet was to adapt its feature extraction more closely to real-world webcam images by training it on the Georgia Tech Face Database, which closely resembles the expected input of our system. However, as mentioned above, the results show that the fine-tuning process had a negligible effect on the

overall performance of the fine-tuned model compared to the original FaceNet model.

These results reflects the difficulty of fine-tuning a large, generalizable model like FaceNet with a small and uniform dataset. The training was intentionally set up to be conservative by keeping all 447 layers of the model frozen except for the final 30, as well as freezing the Batch Normalization layers. This was done to avoid overfitting because of the small dataset while still trying to leave room for adaptation. Since we also opted for a small learning rate and few epochs to avoid catastrophic forgetting of previously learned features, it likely considerably limited the potential for the model to make any significant improvements.

Balancing overfitting against the need for meaningful improvements proved challenging. The use of semi-hard triplet was intended to maximize learning from difficult examples without overwhelming the model with noise or trivial examples. Still, the lack of improvements suggests that either the dataset was too limited in scope or the model already generalized well enough to these types of images, which would leave little room for improvement through fine-tuning alone.

There are also alternative strategies that could have been more effective. One such method could have been to freeze the entire FaceNet model and instead train a custom classification layer on top of it using the Georgia Tech data. This would allow the model to map feature vectors to identities tailored specifically toward webcam images, without disrupting the robust and generalizable features of the original FaceNet model, and also not be bound to only linear evaluation like euclidean distance.

Collecting a larger and more diverse dataset of webcam images, ideally in the same environment and lighting as intended for our system, would allow for more aggressive fine-tuning of the model to enable meaningful improvements where our current results were flat. Although our fine-tuning approach was implemented correctly and with sensible reasoning, it highlighted the limitations of adapting a sophisticated, high-performing model with limited data. To achieve more notable improvements, either the training scope must expand, or the original model should be left unchanged and instead extended with lightweight specialization techniques, rather than retraining existing layers.

7.3 Ethical Considerations

FR systems involve multiple ethical considerations, from the construction of training datasets and model development to their real-world applications. To ensure personal security for those whose data is collected, as well as high performance with minimal bias, these aspects must be thoroughly evaluated both before and during the implementation of such systems.

To begin with, the models require large amounts of data to function effectively. Therefore, the datasets used must contain a substantial number of images along

with relevant metadata. Since this data can be classified as sensitive information, it is crucial that individuals providing it are aware of and consent to its usage. This is regulated by the General Data Protection Regulation (GDPR) [36]. With the growing use of social media, the possibility to create extensive datasets of facial images has increased, raising important ethical questions about how this data can be used. This project relies on pre-existing datasets which means that the responsibility of consent lay on the creators of the datasets. However, we handle personal information for the login credentials in our web application which entails responsibility for handling this storage. To make sure that the user always can withdraw their consent to the storage of data in their account we implemented the functionality for deleting the account. This option removes all data connected to the user from the database which to the most basic level meets the requirements of GDPR. Despite our attempts of improving this, if this were to be developed to a more large scale application, informational sections of how the data is stored and more formal questions for the user to consent to it would need to be developed.

The diversity of images in the datasets is also an important ethical consideration. Depending on how the dataset is constructed, biases can easily arise in the models, potentially leading to unfair treatment of certain user groups. Wanyonyi et al. discuss this issue and conclude that using homogeneous datasets can reduce model performance and contribute to demographic bias, ultimately leading to discrimination [4]. As this project and produced prototype won't be used in a large scale no vulnerable user groups will be directly discriminated. For further, more large scale implementations it is important to take this cause of biases in consideration. Multiple datasets have been constructed more diversely to address this problem and such datasets would be preferable when training a new model.

Another important consideration is that objects and garments can interfere with the detection and verification steps of the FR pipeline. Individuals wearing glasses, hats, or religious garments such as burqas may be more difficult to detect and identify compared to those without such items. To address this, detection methods with high accuracy despite occlusion has been used.

Lastly, the use and implementation of FR models involve important ethical considerations. The ethical impact varies depending on how these models are applied. To protect users' privacy and security, appropriate regulations must be established. However, the wide range of possible implementations makes designing such regulations challenging, but nonetheless important. The scope of this project excludes some of these issues but for further implementations it is important to take into consideration.

7.4 Areas for Future Research

Although this project demonstrates the potential of FR for secure web authentication, several aspects remain open for further development. Future work could focus on strengthening spoofing defenses, expanding the system into an authentication

provider, and exploring additional security and usability features. The following sections highlight possible directions for continued research.

7.4.1 Anti-Spoofing Measures

To improve security, this project used a model specifically trained for anti-spoofing. However, that model mainly focused on attacks based on a user holding a picture of another person. This means that the model is not trained to handle spoofing attacks based on other methods, such as 3D-models, high-quality video replays, or deep fakes.

Future research could focus on making further steps to improve the existing FR models. This could be done by performing transfer learning on more comprehensive datasets. But this can also be done by researching advanced liveness detection techniques, such as eye movement tracking, challenge-response mechanisms, or multimodal biometric authentication (e.g., combining FR with voice or fingerprint verification). It would also be possible to explore the use of additional hardware specifically suited for biometrics, since an increasing number of computers on the markets are equipped with depth-sensing cameras or finger print login.

7.4.2 API Authentication Service Extension

Authentication via third-party providers is becoming increasingly standard in modern web applications. For instance, it is standard to see options like "Sign in with Google" on most platforms. A future suggestion would be to dive deeper into how these systems are developed and extend the application into an authentication service. This would involve implementing an OAuth 2.0-compliant authorization server or a custom authentication API that external developers can integrate to offer our service as an authentication option in their applications. In such a scenario, users would be able to authenticate across the web using FR, offering a user-friendly and passwordless experience.

However, building and maintaining such a system would require substantial infrastructure and investment. The costs associated with secure data storage, real-time face recognition, up-time reliability, and regulatory compliance could be significant.

7.4.3 Scalability and Real-World Deployment

The system developed in this project functions as a proof-of-concept for web-based biometric authentication. While the core functionality has been demonstrated in a controlled environment, it has not been evaluated under real-world conditions. As a result, several limitations remain unaddressed. For instance, factors such as server load, latency, cross-device compatibility, and performance under varying network conditions could significantly impact the system's reliability and user experience. Future work could involve comprehensive testing and optimization to assess how the system performs in realistic usage scenarios.

8

Conclusion

To explore biometric authentication as a viable alternative to traditional password-based authentication, we developed a proof-of-concept web application utilizing open-source models for face recognition (FR), face detection (FD), and face anti-spoofing (FAS).

Our FR system is user-friendly, responsive and demonstrates a promising level of performance and usability, with accuracy scores reaching human-level accuracy. Trade-offs between precision and recall were observed in both the recognition and anti-spoofing components, highlighting the challenge of balancing decision thresholds to meet both security and user convenience requirements. As a result, the system owner may choose to prioritize either minimizing the risk of granting access to illegitimate users or minimizing the risk of denying access to legitimate users by lowering or raising the threshold, respectively. The threshold can also be set to maximize average performance metrics, resulting in the best overall balance between security and accessibility. However, optimizing the threshold for the FAS model resulted in significantly lower performance metrics compared to the FR model, regardless of the chosen optimization direction. Therefore, we recommend disabling the FAS component in its current form.

Fine-tuning the FR model through transfer learning had a negligible impact on recognition accuracy, likely due in part to the model's generality. To achieve better results, new high-quality training data must be obtained, and alternative transfer learning strategies should be explored.

In conclusion, the proof-of-concept web application demonstrates the technical viability of FR for user authentication in a web environment and serves as a solid foundation for future development. Future research includes improving FAS detection to achieve higher performance metrics, as well as extending these measures to address a wider range of spoofing attacks. An API authentication service to enable wide adoption and consequently real-world scalability testing to assess whether the concept would successfully scale with a large user base. If developed responsibly with ethical considerations mindful of privacy regulations, biometric authentication systems can meaningfully improve both the security and user experience of digital authentication as an alternative to traditional password-based authentication.

Bibliography

[1] Stöldskyddsföreningen. "Stor ökning av kapade konton i sociala medier." (2024), [Online]. Available: https://www.stoldskyddsforeningen.se/dataintrang-tips/ (Accessed: 01/31/2025).

[2] M. Cooney. "Kostnaden för dataintrång bara stiger – största ökningen sedan pandemin." (2024), [Online]. Available: https://computersweden.se/article/3479480/kostnaden-for-dataintrang-bara-stiger-storsta-okningen-sedan-pandemin.html (Accessed: 01/31/2025).

[3] H. Wang, K. Fan, H. Chen, Q. Chen, K. Zhang, F. Li, H. Li, and Y. Yang, "Joint Biological ID : A Secure and Efficient Lightweight Biometric Authentication Scheme," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 3, pp. 2578–2592, 2023. DOI: 10.1109/TDSC.2022.3186999.

[4] D. Wanyonyi and T. Celik, "Open-source face recognition frameworks: A review of the landscape," *IEEE Access*, vol. 10, pp. 50 601–50 623, 2022. DOI: 10.1109/ACCESS.2022.3170037.

[5] Y. Kortli, M. Jridi, A. Al Falou, and M. Atri, "Face Recognition Systems: A Survey," *Sensors*, vol. 20, no. 2, p. 342, Jan. 2020, ISSN: 1424-8220. DOI: 10.3390/s20020342. [Online]. Available: https://www.mdpi.com/1424-8220/20/2/342 (Accessed: 01/31/2025).

[6] P. Durai, *Face recognition models: Advancements, toolkit, and datasets*, May 2023. [Online]. Available: https://learnopencv.com/face-recognition-models/ (Accessed: 01/30/2025).

[7] A. Kumar, A. Kaur, and M. Kumar, "Face detection techniques: A review," *Artificial Intelligence Review*, vol. 52, no. 2, pp. 927–948, Aug. 2019, ISSN: 1573-7462. DOI: 10.1007/s10462-018-9650-2. [Online]. Available: https://doi.org/10.1007/s10462-018-9650-2 (Accessed: 03/18/2025).

[8] R. Sheldon, N. Barney, and B. Corinne, *What Is Face Detection and How Does It Work? | Definition from TechTarget*. [Online]. Available: https://www.techtarget.com/searchenterpriseai/definition/face-detection (Accessed: 03/18/2025).

[9] S. M. Arsath, *Neural Networks Explained: Inside the Black Box*, Jun. 2024. [Online]. Available: https://medium.com/@arsathcomeng/neural-networks-explained-inside-the-black-box-525c39677edc (Accessed: 03/18/2025).

[10] H. Wang, J. Hu, and W. Deng, "Face Feature Extraction: A Complete Review," *IEEE Access*, vol. 6, pp. 6001–6039, 2017, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2017.2784842. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8225635 (Accessed: 04/20/2025).

[11] X. Wang, J. Peng, S. Zhang, B. Chen, Y. Wang, and Y.-H. Guo, "A Survey of Face Recognition," *ArXiv*, vol. abs/2212.13038, 2022. DOI: 10.48550/arXiv.2212.13038. [Online]. Available: https://api.semanticscholar.org/CorpusID:255125348 (Accessed: 03/08/2025).

[12] B. Amirgaliyev, M. Mussabek, T. Rakhimzhanova, and A. Zhumadillayeva, "A Review of Machine Learning and Deep Learning Methods for Person Detection, Tracking and Identification, and Face Recognition with Applications," *Sensors*, vol. 25, no. 5, p. 1410, Feb. 2025, ISSN: 1424-8220. DOI: 10.3390/s25051410. [Online]. Available: https://www.mdpi.com/1424-8220/25/5/1410 (Accessed: 03/21/2025).

[13] OAIC, *Facial recognition technology: A guide to assessing the privacy risks*, Nov. 2024. [Online]. Available: https://www.oaic.gov.au/privacy/privacy-guidance-for-organisations-and-government-agencies/organisations/facial-recognition-technology-a-guide-to-assessing-the-privacy-risks (Accessed: 03/23/2025).

[14] E. Vezzetti, *Similarity Measures for Face Recognition*. Oak Park: Bentham Science Publishers, 2015, pp. 73–79, ISBN: 9781681080444.

[15] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2021. DOI: 10.1109/JPROC.2020.3004555.

[16] M. Coşkun, A. Uçar, Ö. Yildirim, and Y. Demir, "Face recognition based on convolutional neural network," in *2017 International Conference on Modern Electrical and Energy Systems (MEES)*, 2017, pp. 376–379. DOI: 10.1109/MEES.2017.8248937.

[17] S. Mukherjee, P. Sen, S. Bora, and C. Pradhan, "SQL Injection: A sample review," in *2015 6th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Jul. 2015, pp. 1–7. DOI: 10.1109/ICCCNT.2015.7395166. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7395166 (Accessed: 03/29/2025).

[18] OWASP, *Input Validation - OWASP Cheat Sheet Series*, n.d. [Online]. Available: https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html (Accessed: 03/29/2025).

[19] J. Määttä, A. Hadid, and M. Pietikäinen, "Face spoofing detection from single images using micro-texture analysis," in *2011 International Joint Conference on Biometrics (IJCB)*, 2011, pp. 1–7. DOI: 10.1109/IJCB.2011.6117510.

[20] Apple Inc., *About Face ID advanced technology*, Dec. 2024. [Online]. Available: https://support.apple.com/en-us/102381 (Accessed: 05/15/2025).

[21] IBM, *What is encryption?* 2025. [Online]. Available: https://www.ibm.com/think/topics/encryption (Accessed: 04/30/2025).

[22] Varun, *What is Face Detection? Ultimate Guide 2025 + Model Comparison*, Sep. 2022. [Online]. Available: https://learnopencv.com/what-is-face-detection-the-ultimate-guide/ (Accessed: 02/20/2025).

[23] A. Stailey-Young, *What's the Best Face Detector?* Jun. 2024. [Online]. Available: https://medium.com/pythons-gurus/what-is-the-best-face-detector-ab650d8c1225 (Accessed: 02/20/2025).

[24] S. Sefik, *DeepFace: A Popular Open Source Facial Recognition Library*, Jan. 2023. [Online]. Available: https://viso.ai/computer-vision/deepface/ (Accessed: 03/08/2025).

[25] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, ISSN: 1063-6919, Jun. 2015, pp. 815–823. DOI: 10.1109/CVPR.2015.7298682. [Online]. Available: https://ieeexplore.ieee.org/document/7298682 (Accessed: 02/26/2025).

[26] M. Aleshin, *Hairymax/Face-AntiSpoofing*, original-date: 2022-07-29T06:52:25Z, May 2025. [Online]. Available: https://github.com/hairymax/Face-AntiSpoofing (Accessed: 03/27/2025).

[27] S. Zhang, Z. Zhu, Z. Lei, H. Shi, X. Wang, and S. Z. Li. "Celeba-spoof: Large-scale face anti-spoofing dataset with rich annotations." (2020), [Online]. Available: https://github.com/DengPingFan/CelebA-Spoof (Accessed: 05/16/2025).

[28] National Institute of Standards and Technology (NIST), *Advanced Encryption Standard (AES)*, Nov. 2001. DOI: https://doi.org/10.6028/NIST.FIPS.197. [Online]. Available: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=901427 (Accessed: 04/30/2025).

[29] Apple Support, *Secure Enclave*, Dec. 2024. [Online]. Available: https://support.apple.com/en-gb/guide/security/sec59b0b31ff/web (Accessed: 04/30/2025).

[30] M. Dworkin, *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC*, Nov. 2007. [Online]. Available: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=51288 (Accessed: 04/30/2025).

[31] The cryptography developers, *Cryptography*, n.d. [Online]. Available: https://pypi.org/project/cryptography/ (Accessed: 05/02/2025).

[32] H. Huan, W. Li, and A. K. Jain, *MSU Mobile Face Spoofing Database (MSU-MFSD)*, 2015. [Online]. Available: https://sites.google.com/site/huhanhomepage/datasetcode (Accessed: 05/15/2025).

[33] A. V. Nefian. "Georgia tech face database." (1999), [Online]. Available: https://www.anefian.com/research/face_reco.htm (Accessed: 03/29/2025).

[34] The Artificial Intelligence Laboratory of FEI. "FEI Face Database." (2023), [Online]. Available: https://fei.edu.br/~cet/facedatabase.html (Accessed: 03/29/2025).

[35] D. Timoshenko, K. Simonchik, V. Shutov, P. Zhelezneva, and V. Grishkin, "Large crowdcollected facial anti-spoofing dataset," in *2019 Computer Science and Information Technologies (CSIT)*, Sep. 2019, pp. 123–126. DOI: 10.1109/CSITechnol.2019.8895208.

[36] Intramed. "Copyright and use of images in an article, thesis summary or poster." (2025), [Online]. Available: https://www.intramed.lu.se/en/phd-studies/networking-support-and-guidance-during-your-studies/publishing-and-preparing-manuscripts/copyright-and-use-images-article-thesis-summary-or-poster?q=phd-studies/networking-support-and-guidance-during-your-studies/publishing-and-preparing-

manuscripts/copyright-and-use-images-article-summary-or-poster (Accessed: 02/02/2025).

A

Appendix 1

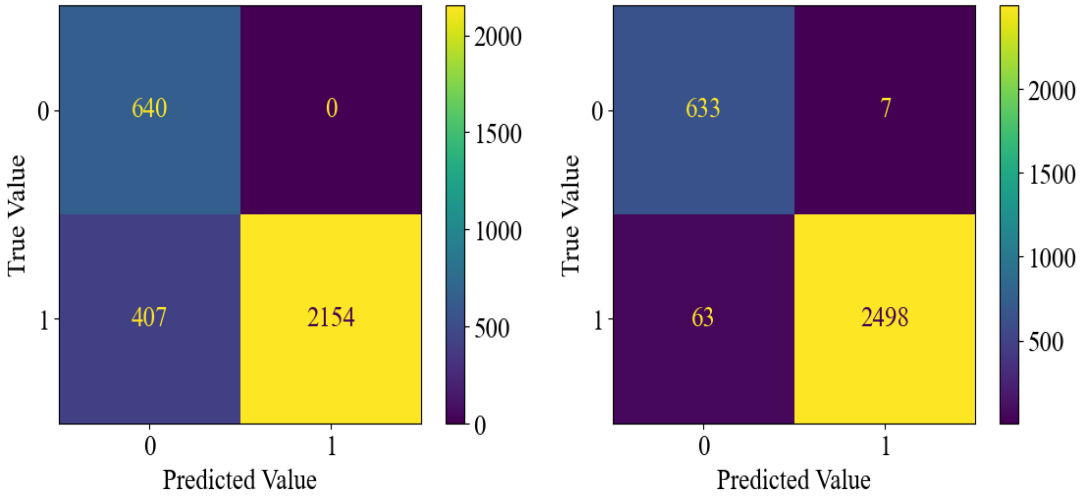

Figure A.1: Confusion matrices over the Final Product FR performance on the FEI dataset, without alignment detection. The thresholds are 0.7 and 1.0, respectively.

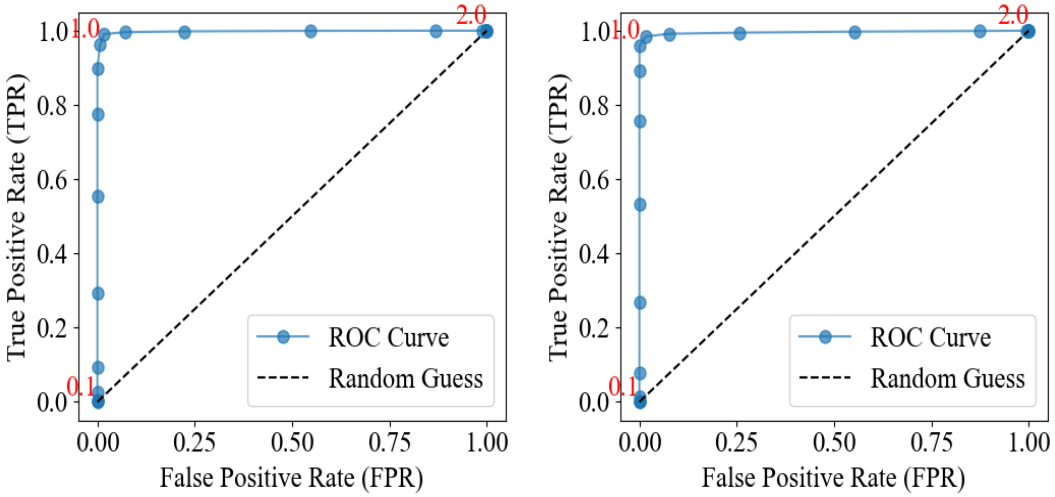

Figure A.2: ROC curves for the Final Product performance test on the LFW dataset with (left curve) and without (right curve) alignment detection.

Table A.1: Performance metrics for the LFW dataset, with alignment detection.

Threshold	Accuracy	Precision	Recall	F1-Score
0.8	0.8205	1.0000	0.7756	0.8736
0.9	0.9194	1.0000	0.8993	0.9470
1.0	0.9694	0.9982	0.9635	0.9806
1.1	0.9892	0.9956	0.9909	0.9933
1.2	0.9826	0.9824	0.9961	0.9892
1.3	0.9542	0.9473	0.9983	0.9721

Table A.2: Performance metrics for the LFW dataset, without alignment detection.

Threshold	Accuracy	Precision	Recall	F1-Score
0.8	0.8060	1.0000	0.7575	0.8620
0.9	0.9145	1.0000	0.8931	0.9435
1.0	0.9682	0.9999	0.9604	0.9797
1.1	0.9844	0.9957	0.9847	0.9902
1.2	0.9780	0.9809	0.9917	0.9863
1.3	0.9444	0.9393	0.9948	0.9663

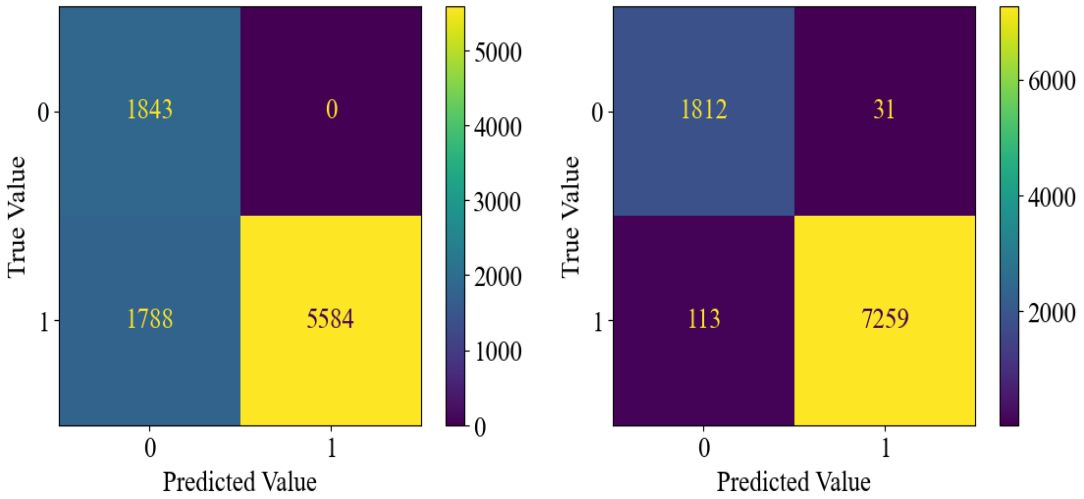

Figure A.3: Confusion matrices over the Final Product FR performance on the LFW dataset, without alignment detection. The thresholds are 0.8 and 1.1, respectively.

B

Appendix 2

User Testing Form

Basic info

Age:
Educational Background:

Tasks

- Register an account
- Change username
- Log out
- Log in
- Delete your account

Questions

- What do you think about the design?
- What felt intuitive/unintuitive in the application?
- Is there anything missing that you would like to point out?
- Would you consider using a similar website with face recognition for authentication in the future? Why/Why not?