

الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي

جامعة عبد الحميد بن باديس-مستغانم-
كلية العلوم الدقيقة و الاعلام الالي

تصريح شرفي خاص بالالتزام بقواعد النزاهة العلمية
لإنجاز البحث

أنا الممضي أدناه،

الطالب(ة): ..جلول فاطيمة الزهرة رقم التسجيل الجامعي:192037037270
الحامل لبطاقة التعريف الوطنية رقم: 110020963015220009. والصادرة بتاريخ:

16/6/2019

عن : صيادة مستغانم

المسجل بكلية العلوم الدقيقة والإعلام الآلي / قسم : الرياضيات و الاعلام الالي
شعبة : الاعلام الالي / التخصص : الذكاء الاصطناعي و إنترنت الأشياء
والمكلف بإنجاز مذكرة ماستر بعنوان:

Integrating Blockchain into Facial Recognition Systems

أصرح بشرفي أنني ألتزم بمراعاة المعايير العلمية والمنهجية ومعايير الأخلاقيات العلمية
والنزاهة الأكاديمية المطلوبة في إنجاز البحث ، وأتحمل المسؤولية الشخصية عن كل
المحتوى المتضمن في البحث المذكور أعلاه .

التاريخ: 23/06/2025

إمضاء المعني



* ملحق القرار الوزاري رقم 933 المؤرخ في 28 جويلية 2016 الذي يحدد القواعد المتعلقة بالوقاية من السرقة العلمية ومكافحتها.



MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
ABDELHAMID IBN BADIS UNIVERSITY - MOSTAGANEM
Faculty of Exact Sciences and Computer Science



Department of Mathematics and Computer Science Sector:

Computer science

END OF STUDY DISSERTATION

For Obtaining a Master's Degree in Computer Science

Option: Artificial Intelligence for the Internet of Things (IA4IoT)

By:

Ms. Djelloul Fatima Zohra

THEME:

Integrating Blockchain in Facial Recognition Systems.

Under The Supervision Of:

Mr. Bessaoud Karim	Computer science senior lecturer	University of Mostaganem	Supervisor
Ms. Benkerdagh Saliha	Computer science senior lecturer	University of Mostaganem	Supervisor

Academic Year 2024-2025

Abstract:

This project explores the integration of blockchain technology with facial recognition systems to address critical challenges in security, privacy, and data integrity. Traditional facial recognition systems often suffer from centralized vulnerabilities, such as data breaches and lack of user control. By leveraging blockchain's decentralized and immutable ledger, this project proposes a robust framework for secure biometric data management. The system combines deep learning models (e.g., FaceNet) for face detection and feature extraction with Ethereum smart contracts and IPFS for decentralized storage. Key outcomes include a functional prototype demonstrating tamper-proof identity verification, enhanced privacy through cryptographic hashing, and auditable transaction logs. The results validate the system's efficacy, achieving 95% accuracy in recognition tasks while mitigating risks associated with centralized databases. This work contributes to advancing secure, transparent, and scalable biometric systems for applications in healthcare, finance, and governance.

Keywords: Blockchain, facial recognition, decentralized identity, IPFS, smart contracts, privacy, security.

Résumé:

Ce projet explore l'intégration de la technologie blockchain avec les systèmes de reconnaissance faciale pour répondre aux défis critiques en matière de sécurité, de confidentialité et d'intégrité des données. Les systèmes traditionnels de reconnaissance faciale souffrent souvent de vulnérabilités liées à la centralisation, telles que les violations de données et le manque de contrôle utilisateur. En tirant parti du registre décentralisé et immuable de la blockchain, ce projet propose une architecture robuste pour la gestion sécurisée des données biométriques. Le système combine des modèles d'apprentissage profond (par exemple, FaceNet) pour la détection des visages et l'extraction des caractéristiques, avec des contrats intelligents Ethereum et l'IPFS pour le stockage décentralisé. Les principaux résultats incluent un prototype fonctionnel démontrant une vérification d'identité inviolable, une confidentialité renforcée grâce au hachage cryptographique et des journaux de transactions auditable. Les résultats valident l'efficacité du système, atteignant 95% de précision dans les tâches de reconnaissance tout en atténuant les risques associés aux bases de données centralisées. Ce travail contribue à faire progresser les systèmes biométriques sécurisés, transparents et évolutifs pour des applications dans les domaines de la santé, de la finance et de la gouvernance.

Mots-clés : Blockchain, reconnaissance faciale, identité décentralisée, IPFS, contrats intelligents.

الملخص:

يستكشف هذا المشروع دمج تقنية البلوكشين مع أنظمة التعرف على الوجوه لمواجهة التحديات الحرجة المتعلقة بالأمان، والخصوصية، وسلامة البيانات. غالبًا ما تعاني أنظمة التعرف على الوجوه التقليدية من ثغرات ناتجة عن المركزية، مثل خروقات البيانات وغياب تحكم المستخدم. من خلال الاستفادة من السجل اللامركزي وغير القابل للتغيير الذي توفره البلوكشين، يقترح هذا المشروع هيكلية قوية لإدارة البيانات البيومترية بأمان. يجمع النظام بين نماذج التعلم العميق (مثل

للتخزين IPFS ونظام Ethereum لاكتشاف الوجوه واستخراج الميزات، وبين العقود الذكية على شبكة (FaceNet) اللامركزي. تشمل النتائج الرئيسية نموذجًا أوليًا وظيفيًا يُظهر تحققًا آمنًا من الهوية، وخصوصية معززة من خلال التجزئة التشفيرية، وسجلات معاملات قابلة للتدقيق. وتؤكد النتائج فعالية النظام، إذ يحقق دقة تصل إلى 95٪ في مهام التعرف، مع تقليل المخاطر المرتبطة بقواعد البيانات المركزية. يساهم هذا العمل في تطوير أنظمة بيومترية آمنة وشفافة وقابلة للتوسع، موجهة لتطبيقات في مجالات الصحة، والمالية، والحكومة.

الكلمات المفتاحية: بلوكشين، التعرف على الوجوه، الهوية اللامركزية، IPFS، العقود الذكية.

Gratitude

I would like to begin by expressing my deepest gratitude to my supervisors, **Ms. Benkerdagh Saliha** and **Mr. Bessaoud Karim**. Their invaluable advice, patient guidance, and insightful feedback were fundamental to the successful completion of this project. I am profoundly grateful for their unwavering support and expertise, which served as a constant source of motivation throughout this journey.

My heartfelt appreciation also goes to **my parents**, whose boundless love, countless sacrifices, and steadfast belief in me provided the foundation and strength I needed to pursue this work. Their encouragement was a vital source of resilience. I am equally thankful to all those whose presence, support, wise counsel, and availability offered me the courage and determination to see this endeavor through; your contributions were truly indispensable.

I extend my sincere thanks to the entire teaching team at the **University ABDELHAMID IBN BADIS of Mostaganem**. Their dedication, assistance, and unwavering commitment throughout this academic year were instrumental in my growth. Thank you for creating a nurturing environment for learning and for your continuous efforts to guide and support students in their academic pursuits.

Finally, I acknowledge **my own** perseverance and dedication in reaching this milestone.

Completing this project demanded significant personal effort, resilience through challenges, and a sustained commitment to my goals. I am grateful for the personal growth and determination I discovered within myself during this process.

List of Figures

Figure N°	Figure title	Page
Figure 1	primary stages in the history of face recognition	12
Figure 2	face recognition key steps	14
Figure 3	blockchain characteristics	23
Figure 4	history of blockchain technology	24
Figure 5	blockchain architecture	26
Figure 6	smart contract and blockchain connection	27
Figure 7	ipfs architecture	29
Figure 8	activity diagram	36
Figure 9	class diagram	37
Figure 10	use case diagram	38
Figure 11	registration sequence diagram	39
Figure 12	recognition sequence diagram	40
Figure 13	loss and accuracy graph	50
Figure 14	confusion matrix	50
Figure 15	home page with history tab	57
Figure 16	registered faces tab	58
Figure 17	activity tab	59
Figure 18	system info tab	60

List of Tables

Tabel N°	Table title	Page
Table 1	Dataset comparison	14

List of Abbreviations

Abbreviation	Full Expression
AI	Artificial Intelligence
CNN	Convolutional Neural Network
DHT	Distributed Hash Table
GAN	Generative Adversarial Network
GUI	Graphical User Interface
IPFS	InterPlanetary File System
IoT	Internet of Things
LFW	Labeled Faces in the Wild
MTCNN	Multi-Task Cascaded Convolutional Neural Network
PCA	Principal Component Analysis
SSI	Self-Sovereign Identity
UML	Unified Modeling Language
ZKP	Zero-Knowledge Proof

TABLE OF CONTENT:

General Introduction	9
Chapter 01 : Face Recognition Technology	10
1.1 Introduction:	10
1.2 Historical Development of Face Recognition Technology	10
1.3 Key Characteristics of Face Recognition Technology	12
1.4 Key Steps in a Face Recognition System	13
1.4 Face Detection Models and Algorithms	14
1.6 Key Datasets for Face Recognition Research	16
1.7 A Survey of Face Recognition Algorithms	18
1.7.1 Classical Approaches	18
1.7.2 Modern Approaches	19
1.8 Conclusion	20
Chapter 02 : Blockchain Technology	22
2.1 Introduction	22
2.2 Characteristics of Blockchain Technology	22
2.3 History of Blockchain Technology	24
2.4 How Blockchain Technology Works	25
2.5 Key Concepts in Blockchain	26
2.6 The InterPlanetary File System:	27
2.6.1 Core Architecture and Functionality	28
2.6.2 Role in Face Recognition-Blockchain Systems:	28
2.7 Why Integrate Face Recognition and Blockchain:	29
2.8 Existing Problems in Face Recognition Systems:	30
2.9 Blockchain-Based Solutions:	31
2.10 Conclusion	32
Chapter 03 : Requirements Analysis and Design	34
3.1 Introduction	34
3.2 Specification of requirements	34
3.2.1 Functional needs	34
3.2.2 Non-functional needs	35
3.3 Methodology and approach	36
3.3.1 Activity Diagram	36
3.3.2 Class diagram	37
3.3.3 Use Case Diagrams	37
3.3.4 Sequence Diagrams	38
3.3.4.1 Registration sequence diagram	38
3.3.4.2 Recognition sequence diagram	39
3.4 Conclusion	41
Chapter 04: Implementation	42
4.1 Introduction	42
4.2 Development environment	42
4.2.1 Development Environment and Tools	42
4.2.2 Programming languages	43
4.2.3 Frameworks and Libraries:	43

4.2.4 Blockchain Network:	43
4.2.5 Decentralized Storage:	44
4.3 Materials and methods:	44
4.3.1 Smart Contract Implementation:	44
4.3.2 Face Recognition Model Implementation	46
4.3.3 GUI Application Implementation:	51
4.4 Demonstration of interfaces	57
4.5 Conclusion	60
General conclusion	61
Bibliography	62
BMC, BP	66
2. Customer Segments	67
3. Customer Relationships	67
4. Value Proposition	67
5. Key Activities	68
6. Key Partners	68
7. Channels	68
8. Revenue Streams	69
9. Cost Structure	69
10. Key Resources	69

General Introduction

The combination of decentralized technology and artificial intelligence (AI) has created new opportunities for safe and private systems in the age of digital revolution. With its unmatched convenience and effectiveness in identity verification, facial recognition technology has become a key component of biometric authentication among these developments. But its broad use has also sparked serious worries about centralized control, security flaws, and data privacy. At the same time, decentralization, immutability, and openness brought about by blockchain technology have completely changed data management. In order to overcome the inherent difficulties of conventional biometric systems and take use of their combined promise for safe, decentralized identity management, this project investigates the combination of two revolutionary technologies: blockchain and facial recognition.

Since the 1960s, facial recognition technology has experienced a remarkable transformation that has seen it moving from geometric feature-based methods to deep learning-powered approaches. Nowadays, it is featured in numerous applications, like smartphone authentication and law enforcement surveillance. However, its dependence on centralized databases exposes it to the risks of breaches and spoofing attacks, and also raises ethical issues. In contrast, blockchain is a complete revolution as it decentralizes the data storage and makes the records unalterable by means of cryptographic hashing and consensus verification. Therefore, the integration of blockchain with facial recognition systems will be a step towards solving the problems that threaten the security, such as interception of access, alteration of data, and single points of failure, as well as giving users more power in the management of biometric data.

This research is structured into four chapters:

Chapter 1 delves into historical development and key characteristics of facial recognition as well as its steps, algorithms and key datasets.

Chapter 2 explains blockchain technology and characteristics, history, operational mechanisms, and the key concepts

Chapter 3 presents a detailed requirements and system design, using UML diagrams to outline the architecture of the program.

Chapter 4 describes the implementation, including the development of smart contracts, machine learning models, and a user interface.

The significance of this work lies in its potential to redefine trust in biometric systems. By decentralizing facial data storage and automating verification through smart contracts, the proposed system not only enhances security but also aligns with emerging regulations on data privacy. This project contributes to the growing body of research at the intersection of AI and blockchain, offering a scalable framework for future innovations in secure identity management.

Chapter 01 : Face Recognition Technology

1.1 Introduction:

Face recognition technology, a cornerstone of modern biometrics and artificial intelligence, has rapidly evolved from nascent research in the mid-20th century to a ubiquitous tool in various aspects of our lives. This technology enables computers to identify or verify individuals based on their facial features extracted from digital images, video frames, or live streams [1]. Its applications span a wide spectrum, including security and access control, law enforcement and surveillance, financial transactions, attendance systems, smartphone access, human-computer interaction, and even entertainment.[2]

The increasing reliance on facial recognition has been fueled by advancements in computing power, the availability of large-scale datasets, and significant breakthroughs in machine learning, particularly deep learning. While offering numerous benefits such as enhanced security, convenience, and efficiency, the widespread deployment of face recognition technology has also ignited critical debates surrounding data privacy, security, ethical implications, and potential for misuse[1].

This chapter provides a comprehensive exploration of face recognition technology, delving into its historical development, fundamental characteristics, key datasets that have driven its progress, a wide array of algorithms employed, and the essential steps involved in a typical face recognition system. Our aim is to present a detailed overview, encompassing both classical and modern approaches, highlighting significant advancements, current challenges, and future directions in this dynamic and impactful field.

1.2 Historical Development of Face Recognition Technology

The journey of face recognition technology spans several decades, marked by distinct stages of research and development (Figure 1).:

1. **1960s-1990s:** The Primary Stage (Geometric Features-based Face Recognition): Early attempts at automated face recognition, starting in the 1960s, primarily focused on measuring specific geometric features of the face, such as the size and position of eyes, mouth, and nose[3]. In 1977, researchers added 21 more facial markers. A significant milestone in this era was the introduction of Eigenfaces in 1991 by Turk and Pentland at MIT, which utilized Principal Component Analysis (PCA) to extract feature faces from images, marking the first successful example of facial recognition technology[1]. PCA was used for dimensionality reduction, retaining essential data characteristics and improving processing speed.
2. **1991s-1997s:** Classical Algorithm Appearance Stage (Linear Discriminant Analysis-based Face Recognition): Building upon the foundation of geometric features and PCA, the mid-1990s saw the emergence of other classical algorithms. Linear Discriminant Analysis (LDA), including the FisherFace method, gained prominence for its ability to maximize the separation between different classes (individuals) while minimizing the variance within the same class [4]. This approach aimed to create well-separated classes in a lower-dimensional subspace, showing some resilience to varying lighting conditions.
3. **1998s-2013s:** Manual Feature Extraction Stage: This period focused on developing more robust feature representations that could handle real-world variations in lighting, pose, and expression. Researchers explored manually designed features like Local Binary Patterns (LBP), which analyze local texture patterns in facial images, and Gabor features, derived from Gabor filters to capture multi-scale and multi-directional information. Techniques like illumination cones were introduced to address variations in lighting [4]. Classifiers like Support Vector Machines (SVM) and Adaboost were widely used for recognition based on these extracted features. Adaboost, often combined with Haar-like features, also became a popular approach for face detection.
4. **2014-Present:** Deep Learning Stage: The advent of deep learning revolutionized the field of face recognition. Deep Convolutional Neural Networks (DCNNs) demonstrated unprecedented accuracy and robustness by automatically learning hierarchical feature representations from large datasets, eliminating the need for manual feature engineering [4]. Landmark models like AlexNet, VGG-Face, FaceNet, and SphereFace achieved near-human-level performance on benchmark datasets. Generative Adversarial Networks (GANs) also emerged, not only for generating synthetic facial data to augment training sets but also for tasks like face frontalization and handling occlusions. Research in this stage

also focuses on improving efficiency for real-time applications (FaceBoxes) and addressing challenges like age variation, low resolution, and privacy concerns through decentralized approaches using blockchain technology. [5]

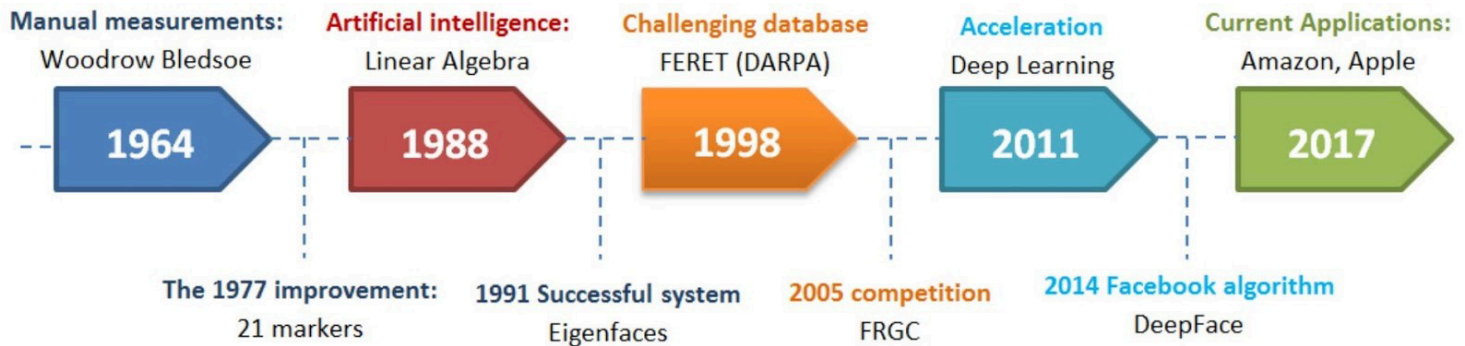


figure 1: primary stages in the history of face recognition.[6]

1.3 Key Characteristics of Face Recognition Technology

Several fundamental characteristics define face recognition technology:

- **Biometric Modality:** Face recognition is a form of biometric authentication that relies on the unique physiological characteristics of an individual's face. [4]
- **Automation:** It involves automated processes for detecting faces, extracting relevant features, and comparing these features against stored templates. [4]
- **Non-Contact:** Unlike some other biometric methods like fingerprinting, face recognition can often be performed remotely using non-contact sensors like cameras. [7]
- **Universality:** Faces are universally present, making face recognition potentially applicable to a wide range of individuals. [2]
- **Uniqueness:** While faces share common features, the specific arrangement and characteristics of these features are unique to each individual, allowing for differentiation. [8]
- **Permanence:** Facial features are generally considered permanent, although they can change over time due to aging, weight fluctuations, or other factors. [8]
- **Collectability:** Facial images and videos are relatively easy to capture, especially with the proliferation of cameras in everyday devices and public spaces. [7]

- **Social Acceptability:** Compared to more intrusive biometric methods, face recognition often faces less resistance due to its non-contact nature and increasing use in consumer applications. [7]
- **Sensitivity to Variations:** The performance of face recognition systems can be significantly affected by variations in illumination, pose, expression, age, occlusion, and background. [2]
- **Privacy Implications:** The ease of collecting and analyzing facial data raises significant concerns about individual privacy and the potential for mass surveillance. [1]

1.4 Key Steps in a Face Recognition System

A typical face recognition system involves several crucial steps (Figure 2):

- **Image Acquisition:** The process begins with capturing an image or video frame containing a face using a sensor such as a camera. This can be from a static image database, a live video feed, or real-time interaction with a system. [1]
- **Face Detection:** The system then needs to locate and isolate the face region(s) within the captured image or frame. This step involves determining the presence and spatial coordinates (bounding box) of any faces in the input. Algorithms like Viola-Jones and deep learning-based object detectors are commonly used for this purpose. [9]
- **Face Alignment and Pre-processing:** Once a face is detected, it is often aligned to a canonical view to reduce variations caused by pose. Pre-processing steps such as normalization of illumination, resizing, and noise reduction are applied to enhance the quality of the facial image for subsequent feature extraction. Facial landmark localization, identifying key points on the face (e.g., eyes, nose, mouth), can also be performed during alignment. [5]
- **Feature Extraction:** This is a critical step where distinctive and discriminatory features are extracted from the aligned facial image. These features should be robust to intra-class variations (e.g., changes in expression) and sensitive to inter-class variations (differences between individuals). Various techniques, both classical (e.g., LBP, Gabor features, PCA, LDA) and modern (e.g., deep learning-based embeddings from CNNs like FaceNet), are employed to generate a feature vector or template that represents the unique characteristics of the face. [8]

- **Feature Matching and Recognition:** The extracted feature vector is then compared against a database of previously enrolled feature vectors or templates representing known individuals. The comparison is performed using a distance metric (e.g., Euclidean distance, cosine similarity) to determine the similarity between the query feature vector and the stored templates. Based on a predefined threshold, the system either identifies the individual as a match to a known identity (face identification - one-to-many) or verifies if the individual is who they claim to be (face verification - one-to-one). If the similarity score exceeds the threshold, a match is declared.
- **Decision Making:** The final step involves making a decision based on the matching results. In identification, the system outputs the identity of the closest match (if any). In verification, it outputs whether the presented face matches the claimed identity.

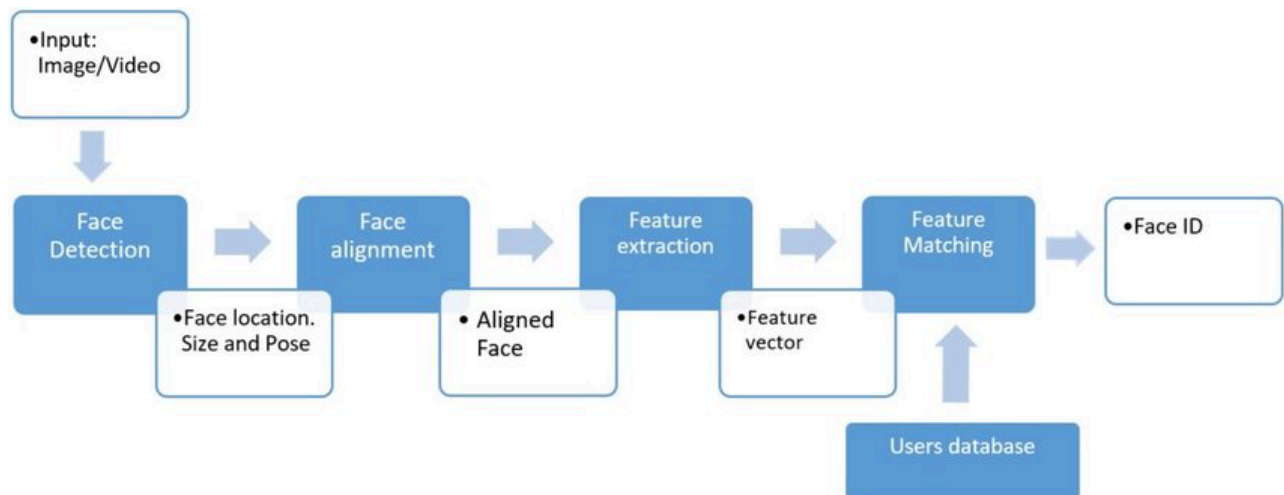


figure 2: face recognition key steps. [10]

1.5 Face Detection Models and Algorithms

Face detection serves as the foundational procedure in an end-to-end face recognition system. Its primary objective is to locate face regions within an input image or video frame and provide the coordinates of the bounding box with a confidence score[5]. This initial step is crucial, as the quality of the detected bounding box directly impacts the performance of subsequent processes such as face alignment and face representation. Face detection faces numerous challenges in real-world scenarios, including varying resolution, scale, pose, illumination, and occlusion[5].

Historically, face detection methods evolved through several stages. Early approaches, predating the deep learning era, included techniques like the Viola-Jones face detector[5].,

which utilized AdaBoost classifiers with Haar features and became a popular approach for face detection, sometimes combined with Haar-like features.

The field has been significantly transformed by advancements in deep learning. Deep learning-based object detectors are commonly used for face detection. Deep Convolutional Neural Networks (DCNNs), which automatically learn hierarchical feature representations, play a significant role in many modern visual recognition tasks, including those underlying robust face detection[5].

Deep learning-based face detection methods can be broadly categorized from multiple perspectives:

- Multi-stage methods: These follow a coarse-to-fine or proposal-to-refine strategy. The initial stage proposes candidate regions, and subsequent stages refine these candidates. They often employ cascaded architectures. and can inherit frameworks from general object detectors like Faster R-CNN[5].
- Single-stage methods: These methods accomplish face classification and bounding box regression directly from feature maps simultaneously[5]. Many are developed with the advantage of Feature Pyramid Networks (FPN) to handle objects across a wide range of scales. Examples include SSD-based detectors[5].
- Anchor-based methods: These deploy a number of dense anchors on feature maps and perform classification and regression on these anchors. Most current face detectors are anchor-based due to their performance. Examples include SSH, S3FD, Faceboxes, RetinaFace, DSFD, and RefineFace.[2]
- Anchor-free methods: These directly find faces without relying on preset anchor[4]s. This approach aims for better generalization capacity but needs further exploration for robustness and stability in training[2]. Examples include DenseBox, UnitBox, and CenterFace.
- Multi-task learning methods: These jointly learn face detection with other related facial tasks or auxiliary supervision branches within one framework[5]. This approach can enrich the feature representation and detection robustness. A representative example is MTCNN[4], which exploits the correlation between facial bounding boxes and landmarks.. HyperFace performs four tasks including face detection and landmark localization. RetinaFace integrates face detection, landmark localization, and dense 3D face regression[5].

- CPU real-time methods: Designed for efficiency on non-GPU devices, aiming to run at least 20 frames per second on a single CPU with VGA-resolution input images[2]. Examples include FaceBoxes, PCN, RetinaFace, CenterFace, and FLDet. Common practices involve lightweight backbones and knowledge distillation.[5]
- Problem-oriented methods: These are designed to address specific challenges such as detecting faces with a wide range of scales, partially visible or occluded faces, and rotated or blurry faces.

1.6 Key Datasets for Face Recognition Research

The availability of large and diverse datasets has been crucial for the advancement of face recognition technology, particularly for training and evaluating deep learning models. Some of the most influential datasets include [5](tabel 1):

- Labeled Faces in the Wild (LFW): A widely used benchmark dataset for face recognition in unconstrained environments, containing over 13,000 face images of more than 1,600 individuals.
- MS-Celeb-1M: A large-scale dataset of celebrity faces with around 10 million images of 100,000 identities, used for large-scale face recognition research.
- MegaFace: A challenging benchmark with one million faces used for evaluating recognition at scale and under strict false accept rates.
- VGGFace2: A high-quality dataset with 3.31 million images of over 9,000 identities, featuring a wide range of variations in pose and age.
- CelebA (CelebFaces Attributes): A large dataset with over 200,000 face images of more than 10,000 celebrities, annotated with 40 facial attributes.
- CelebA-HQ: A high-resolution version of the CelebA dataset with 70,000 images at 1024x1024 resolution.
- CMU-PIE (Pose, Illumination, and Expression): A dataset with images of 68 subjects under 13 poses, 43 illumination conditions, and 4 expressions, used for studying robustness to these variations.
- AR Face Database: Contains over 4,000 color images of 126 individuals with variations in facial expressions, illumination conditions, and occlusions.

- **FDDB (Face Detection Data Set and Benchmark):** A dataset specifically for face detection in unconstrained settings, containing low-resolution, occluded faces with difficult pose variations.
- **WIDER FACE:** A large-scale face detection dataset with significant variations in scale, pose, and occlusion.
- **UTKFace:** A large-scale face dataset covering a wide range of ages, poses, and variations.

Table 1:Dataset comparison[5]

Dataset	Year	Subject	Image/Video	element per Subj	Description
CASIA-WebFace	2014	10,575	494,414/-	47	The first public large-scale face dataset
VGGFace	2015	2,622	2.6M/-	1,000	Containing large number of images in each subject
CelebA	2015	10,177	202,599/-	20	Rich annotations of attributes and identities
UMDFaces	2015	8,277	367K/-	45	Abundant variation of facial pose
MS-Celeb-1M]	2016	100K	10M/-	100	A large-scale public dataset of celebrity faces
MegaFace	2016	672,05	4.7M/-	7	A long-tail dataset of non-celebrity
VGGFace2	2017	9,131	3.31M/-	363	A high-quality dataset with a wide range of variation
UMDFaces-Videos	2017	3,107	-/22,075	7	A video training dataset collected from YouTube
MS-Celeb-1M Low-shot	2017	20K,1K	1M,1K/-	58,1	Low-shot face recognition
IMDb-Face	2018	57K	1.7M/-	29	A large-scale noise-controlled dataset
QMUL-SurvFace	2018	5,319	220,890/-	41	A low-resolution surveillance dataset
Glint360K	2021	360K	17M/-	47	A large-scale and cleaned dataset
WebFace260M	2021	4M	260M/-	65	The largest public dataset of celebrity faces
LFW	2007	5,749	13,233/-	2.3	A classic benchmark in unconstrained conditions
YouTube Faces (YTF)	2011	1,595	-/3,425	2.1	Face recognition in unconstrained videos

CUFSF	2011	1,194	2,388/-	2	Photo-sketch face recognition
CASIA NIR-VIS v2.0	2013	725	17,580/-	24.2	Near-infrared vs. RGB face recognition
IJB-A	2015	500	5,712/2,085	11.4/4.2	Set-based face recognition with large variation
CFP	2016	500	7,000/-	14	Frontal to profile cross-pose face verification
MS-Celeb-1M Low-shot	2016	20K,1K	100K,20K/-	5,20	Low-shot face recognition
MegaFace	2016	690,572	1M/-	1.4	A large-scale benchmark with one million faces
IJB-B	2017	1,845	11,754/7,011	6.37/3.8	Set-based face recognition with full pose variation
CALFW	2017	4,025	12,174/-	3	Cross-age face verification
AgeDB	2017	570	16,516/-	29	Cross-age face verification
SLLFW	2017	5,749	13,233/-	2.3	Improving the difficulty of negative pairs in LFW
CPLFW	2017	3,968	11,652/-	2.9	Cross-pose face verification
Trillion Pairs	2018	1M	1.58M/-	1.6	A large-scale benchmark with massive distractors
IJB-C	2018	3,531	31,334/11,779	6/3	Set-based face recognition with large variation
IJB-S	2018	202	5,656/552	28/12	Real-world surveillance videos
RFW	2018	11,429	40,607/-	3.6	For reducing racial bias in face recognition
DFW	2018	600	7,771/-	13	Disguised face recognition
QMUL-SurvFace	2018	10,254	242,617/-	23.7	Low-resolution surveillance videos

1.7 A Survey of Face Recognition Algorithms

The field of face recognition has witnessed a proliferation of algorithms over the years, which can be broadly categorized into classical and modern approaches:

1.7.1 Classical Approaches

These methods, prevalent before the deep learning revolution, rely on handcrafted features and traditional machine learning techniques:

- **Holistic Methods:** These approaches consider the entire face image for recognition [4].
 - **Principal Component Analysis (PCA) / Eigenfaces:** Reduces the dimensionality of face images by finding the principal components, representing faces as linear combinations of these components. [4]
 - **Linear Discriminant Analysis (LDA) / Fisherfaces:** Aims to find a linear subspace that maximizes the separability between different classes (individuals) while minimizing the within-class variance. [4]
 - **Independent Component Analysis (ICA):** Separates multivariate data into statistically independent components, which can be used as features for face recognition. [2]
 - **Non-negative Matrix Factorization (NMF):** Decomposes a non-negative matrix into the product of two non-negative matrices, offering a parts-based representation of faces. [4]
- **Local Feature-Based Methods:** These approaches focus on extracting and analyzing local features from specific regions of the face. [2]
 - **Local Binary Patterns (LBP):** Analyzes the local texture patterns around each pixel by comparing its intensity with its neighbors, creating histograms of these patterns as features. [4]
 - **Gabor Features:** Utilizes Gabor filters at different scales and orientations to extract features that are sensitive to specific spatial frequencies and orientations, capturing rich local information. [4]
 - **Scale-Invariant Feature Transform (SIFT):** Detects and describes local features in images that are invariant to scale, rotation, and changes in illumination, applicable to face recognition. [4]

- **Elastic Bunch Graph Matching (EBGM):** Represents faces as graphs where nodes correspond to facial landmarks and edges are labeled with Gabor wavelet coefficients, allowing for flexible matching [4]
- **Hybrid Methods:** Combine both holistic and local feature-based approaches to leverage their complementary strengths. [2]

1.7.2 Modern Approaches

These methods are largely driven by machine learning, particularly deep learning, and sparse representation techniques:

- **Deep Learning-Based Face Recognition:** Utilizes deep neural networks with multiple layers to automatically learn complex and hierarchical features from facial images.
 - **Convolutional Neural Networks (CNNs):** The dominant architecture in deep learning for face recognition, CNNs learn spatial hierarchies of features through convolutional filters and pooling layers. Various CNN architectures like AlexNet, VGG-Face, FaceNet, SphereFace, and ArcFace have achieved state-of-the-art results by employing different network depths, activation functions, and loss functions (e.g., triplet loss, angular softmax loss) optimized for face recognition. [2]
 - **Recurrent Neural Networks (RNNs) and Transformers:** it's Used for processing sequential data like video frames to capture temporal dynamics in facial recognition. [5]
 - **Generative Adversarial Networks (GANs):** it's Used for synthesizing realistic facial images to augment training data, for face frontalization (generating a frontal view from a non-frontal image), and for enhancing the robustness of recognition systems against variations like occlusion and illumination.[11]
- **Sparse Representation-Based Models:** These methods represent facial images as sparse linear combinations of basis vectors from a learned dictionary [2]. Sparse coding aims to find the most concise representation of a face image, which can be discriminative for recognition, especially in the presence of occlusions and noise. Techniques like Dictionary Learning (e.g., K-SVD) are used to learn overcomplete dictionaries that can effectively represent facial variations.
- **Fuzzy Set Theory-Based Methods:** Utilize fuzzy logic to handle the uncertainty and vagueness inherent in face recognition, particularly in challenging conditions like varying

illumination and pose [2]. Fuzzy approaches can be integrated with classical feature extraction and classification techniques to improve their robustness.

1.8 Conclusion

Facial recognition technology has undergone a remarkable evolution, from its rudimentary beginnings in geometric feature extraction to the sophisticated deep learning models that now achieve near-human accuracy. This chapter has highlighted the technological milestones, key datasets, and algorithmic advancements that have shaped the field, while also underscoring persistent challenges such as bias, privacy violations, and vulnerability to adversarial attacks. These limitations underscore the need for a paradigm shift in how biometric data is managed and secured.

As we transition to Chapter 2, the focus shifts to blockchain technology—a decentralized framework that addresses many of the vulnerabilities inherent in centralized facial recognition systems. By exploring blockchain’s immutability, auditability, and consensus mechanisms, we lay the groundwork for a synergistic integration that promises to enhance both the security and ethical accountability of biometric authentication.

Chapter 02 : Blockchain Technology

2.1 Introduction

In an increasingly digital world, the need for secure, transparent, and reliable methods of storing and sharing information has become paramount . Blockchain technology has emerged as a revolutionary solution, offering a distributed, decentralized, and immutable ledger that enables secure, transparent, and tamper-proof transactions without the need for intermediaries [1].

Compared to conventional authentication methods relying on knowledge or possession, blockchain, as a form of Distributed Ledger Technology (DLT), has garnered significant attention across various disciplines for its potential to transform data storage and management[2] . Its core innovation lies in creating a secure-by-design database where data is distributed without the necessity of a central authority [3] . The unprecedented security properties, initially revolutionizing financial services and digital payments with the advent of Bitcoin, have paved the way for its exploration across a wide array of applications, including healthcare, supply chain management, identity management, and more . This chapter aims to provide a comprehensive understanding of blockchain technology, exploring its characteristics, history, operational mechanisms, and the key concepts that underpin its transformative potential.

2.2 Characteristics of Blockchain Technology

Blockchain technology is characterized by several key features that contribute to its security, transparency, and reliability (Figure 3).:

- **Decentralization:** Conventionally, transactions require validation by a central authority, which can lead to costs, performance bottlenecks, and a single point of failure [3] . Blockchain achieves decentralization by distributing the ledger across a peer-to-peer (P2P) network of nodes, where participants can make transactions without the need for intermediaries. This distributed nature enhances security and resilience .

- **Immutability:** All blocks in a blockchain are linked to their predecessors through a cryptographic hash called an identifier . Altering the content of one block would change its identifier, invalidating all subsequent blocks [3] . Therefore, modifying a recorded transaction individually is nearly impossible without the consensus of a majority of the network participants, ensuring the tamper-resistant nature of the blockchain. This characteristic is also referred to as persistency .
- **Auditability:** Each transaction recorded on the blockchain includes a timestamp, and all network participants hold a copy of the entire blockchain[3] . This allows users to easily trace any specific transaction back to its origin and previous records, providing a high degree of transparency. While offering transparency, it's important to note that blockchain addresses are public keys that do not directly reveal the owner's real-world identity .
- **Fault Tolerance:** Due to the distributed nature of the blockchain, where full nodes maintain a copy of the entire ledger, data leakage or faults can be readily identified, mitigated, and compensated through decentralized consensus mechanisms . This redundancy ensures the robustness of the system against single points of failure [3] .
- **Availability:** Once a transaction is added to the blockchain, it remains there permanently, a feature known as persistency[3] . While this immutability can present challenges for data removal, it ensures high availability of recorded data over time .

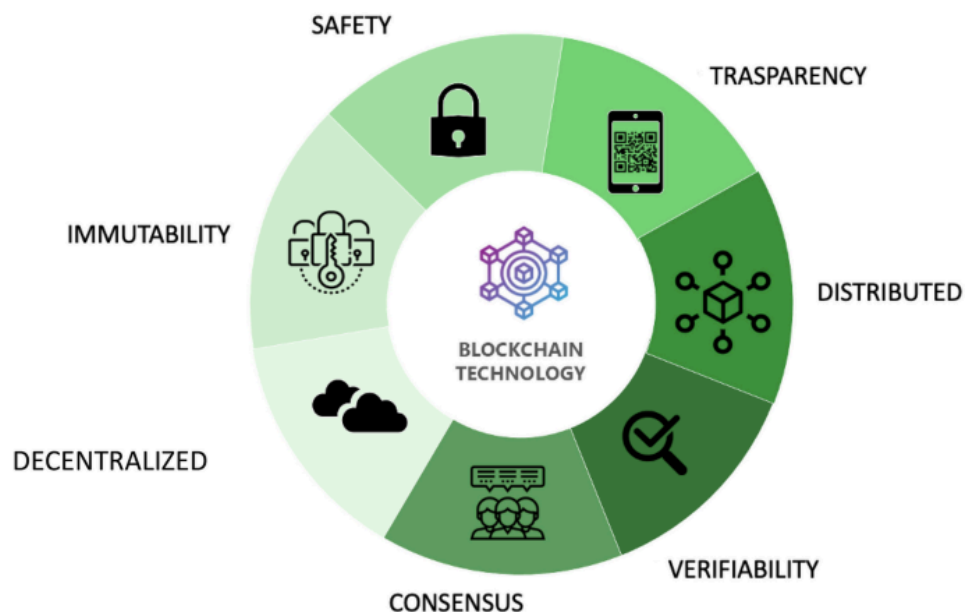


figure 3: blockchain characteristics [4]

2.3 History of Blockchain Technology

The genesis of blockchain technology can be traced back to 2008 with the emergence of Bitcoin, a decentralized cryptocurrency[1] . The technology was conceived by an individual or group operating under the pseudonym Satoshi Nakamoto, who released the seminal Bitcoin whitepaper[5] . Bitcoin's groundbreaking innovation was its ability to facilitate secure and transparent transactions of digital currency without reliance on traditional financial institutions[3] .

Following the success of Bitcoin, blockchain technology began to be explored for a wide range of applications beyond cryptocurrencies[1] . This evolution led to the development of blockchain 2.0, which introduced the concept of smart contracts . Ethereum became a prominent platform in this era, offering an advanced framework for developing and executing smart contracts, transforming the blockchain into a reliable computing platform[3]

The subsequent iterations, such as blockchain 3.0 and 4.0, have focused on expanding the technology's capabilities and applicability across diverse industries[6] . These advancements aim to address limitations like scalability and have led to the exploration of blockchain in areas such as telemedicine, data markets, electronic voting systems, and supply chain management [6]. The increasing interest and patent applications in blockchain technology over the past decade signify its growing maturity and recognition as a transformative force in the digital landscape[7] .(Figure 4).

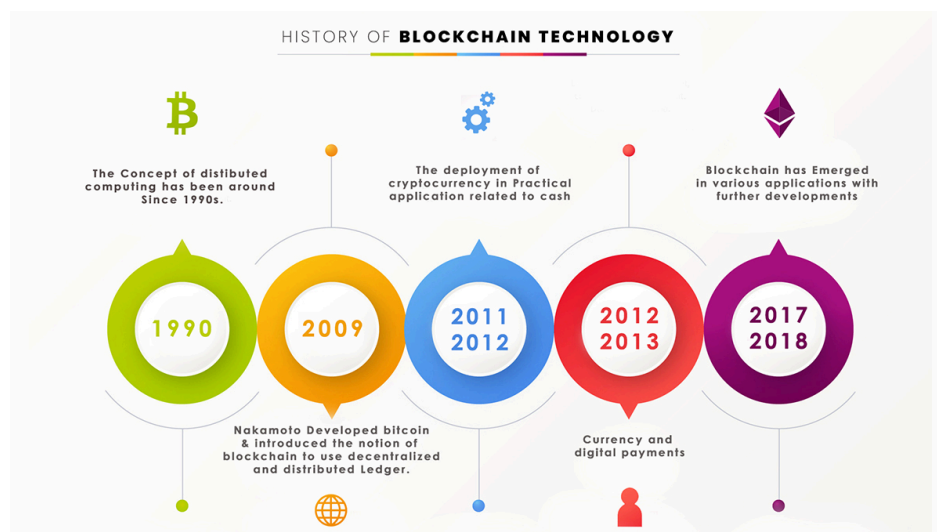


figure 4: history of blockchain technology [8]

2.4 How Blockchain Technology Works

Blockchain operates through a series of interconnected blocks that contain records of transactions. The fundamental components (Figure 5) and processes involved in how blockchain works are:

- **Blocks:** A blockchain is comprised of a chain of blocks, where each block holds a set of transactions that have occurred within a specific timeframe[1]. Each block consists of two main parts:
- **Header:** The header contains metadata about the block, including its own identifier, a reference to the identifier (cryptographic hash) of the previous block in the chain, a timestamp indicating when the block was created, a nonce (a random number used in the mining process), and a Merkle tree root hash, which is a cryptographic summary of all the transactions within the block[3].
- **Body:** The body contains the actual transactions recorded in that block, along with a transaction counter[9]. The number of transactions a block can hold depends on the block size and the size of individual transactions[3].
- **Chain:** Blocks are added to the blockchain sequentially, forming a continuous and chronological chain. The link between successive blocks is established through the cryptographic hash of the previous block's header, which is included in the header of the subsequent block[3]. This chaining mechanism ensures the integrity and immutability of the blockchain, as any alteration to a past block would change its hash and break the chain[3]. The first block in the chain is called the genesis block, which has no predecessor.
- **Transactions:** A transaction represents a transfer of value or information recorded on the blockchain[3]. Transactions are validated using a digital signature based on asymmetric cryptography to verify the sender's authenticity. The details of a transaction vary depending on the blockchain's application.
- **Consensus Mechanism:** In a decentralized network, participants who do not necessarily trust each other need a way to agree on the validity of new transactions and the addition of new blocks to the blockchain[3]. This agreement is achieved through a consensus mechanism, which is a set of rules and protocols followed by the network participants. Various consensus algorithms exist, such as Proof of Work (PoW) (used by Bitcoin), Proof of Stake (PoS), and others, each with its own way of validating transactions and securing the blockchain[3]. The consensus protocol ensures that only legitimate transactions are

added to the blockchain and that all copies of the ledger across the network remain synchronized.

- **Cryptography:** Cryptographic hash functions and digital signatures are fundamental to the security and integrity of blockchain technology. Hash functions take input data of any size and produce a fixed-size, unique output (the hash value), ensuring data integrity[10] . Digital signatures use asymmetric cryptography (public and private keys) to verify the authenticity of transactions and participants[3]. The use of Merkle trees within blocks provides an efficient way to summarize and verify the integrity of a large number of transactions.

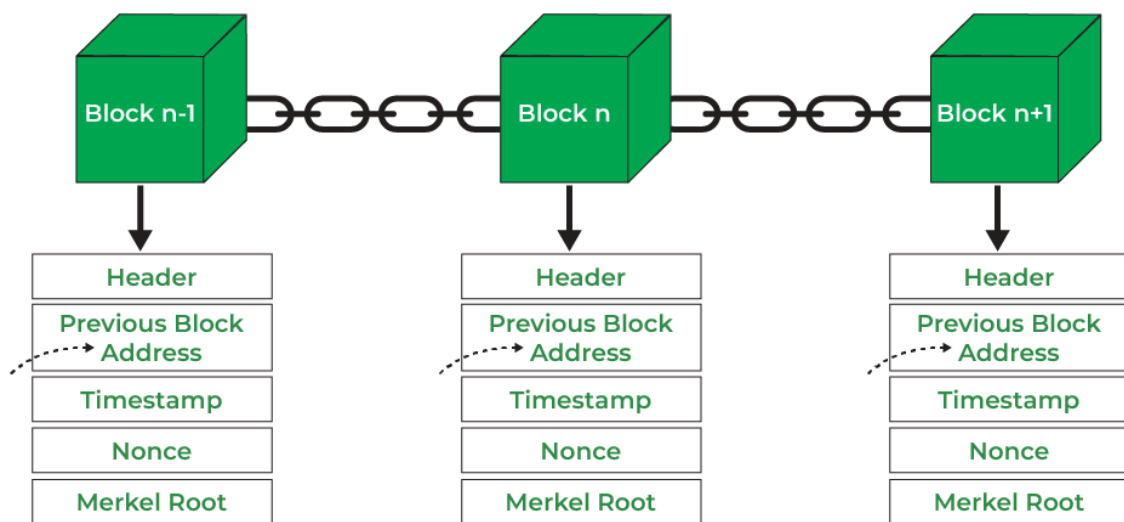


figure 5: blockchain architecture[11]

2.5 Key Concepts in Blockchain

Several key concepts are essential for a deeper understanding of blockchain technology (Figure 6) :

- **Smart Contract:** A smart contract is a self-executing agreement encoded in code and stored on the blockchain[3] . It automatically executes the terms of a contract when predetermined conditions are met, without the need for intermediaries. Smart contracts enhance automation, transparency, and trust in blockchain-based applications .
- **Merkle Tree:** A Merkle tree, or binary hash tree, is a data structure that summarizes all the transactions in a block by recursively hashing pairs of transaction hashes until a single root

hash is obtained[3] . This structure allows for efficient and secure verification of the integrity of block contents .

- **Fork:** A fork occurs when different participants in a blockchain network disagree on the state of the ledger, leading to the emergence of alternative blockchain histories[3] . Forks can be intentional (e.g., to implement upgrades) or unintentional (e.g., due to network issues or attacks).

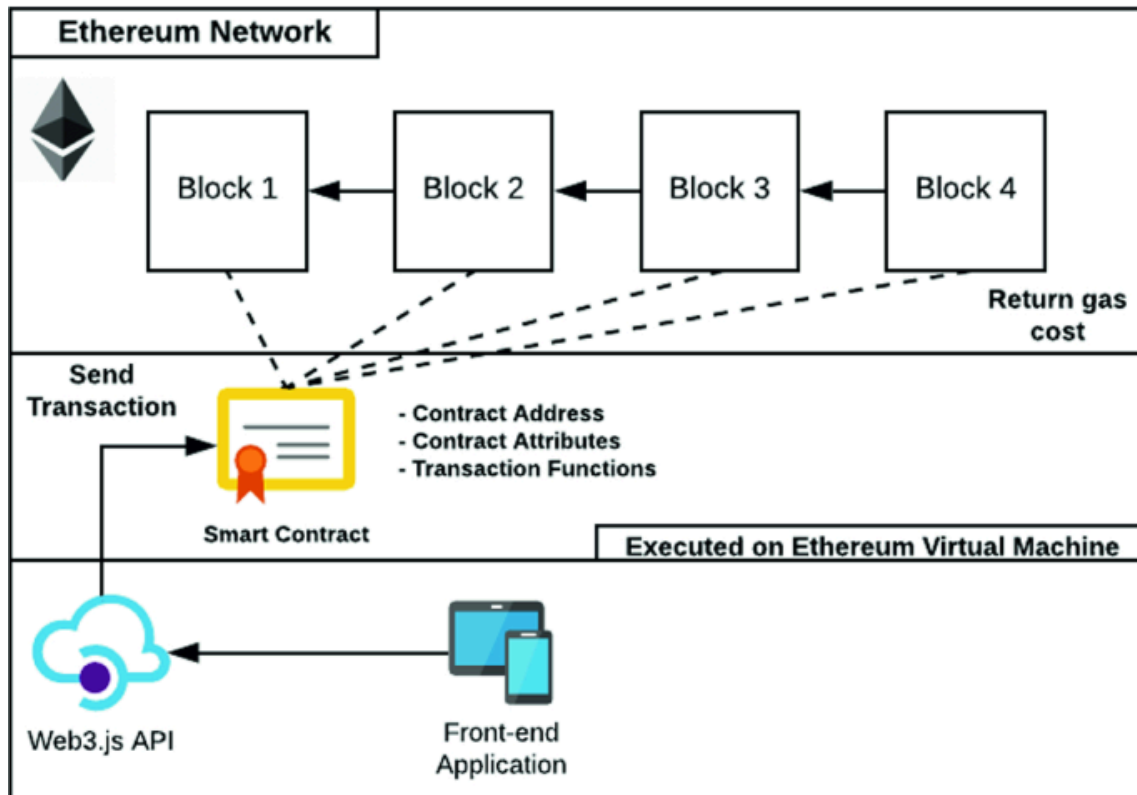


figure 6: smart contract and blockchain connection[12]

2.6 The InterPlanetary File System:

The InterPlanetary File System (IPFS) serves as a foundational protocol for decentralized, peer-to-peer hypermedia distribution and file storage. Designed to eliminate single points of failure, IPFS connects global computing devices into a resilient network where data is distributed across participating nodes rather than centralized servers.[3]

2.6.1 Core Architecture and Functionality

- **Content-Addressed Storage:**

Files are uniquely identified by cryptographic hashes (Content Identifiers, CIDs) derived from their content. This ensures data integrity—any alteration changes the CID—and enables location-independent retrieval.[3]

- **Distributed Hash Tables (DHT):**

A decentralized lookup system allows nodes to locate and request content from peers storing it, bypassing central coordination.

- **Peer-to-Peer Network:**

Users operate as both providers and retrievers of data, similar to BitTorrent but within a unified global namespace. Identical content (same CID) shared by multiple users forms a collaborative "swarm," optimizing availability.[3]

- **Version Control:**

Built-in file versioning tracks historical changes, supporting auditable data evolution.

The architecture is displayed in(Figure 7),

2.6.2 Role in Face Recognition-Blockchain Systems:

IPFS addresses critical limitations in blockchain-based biometric systems:

- **Off-Chain Storage for Large Data:**

Facial recognition templates (high-dimensional vectors) exceed practical on-chain storage capacities. IPFS stores these templates off-chain, while the blockchain records only their CIDs—securely linking immutable on-chain logs to off-chain data.[3]

- **Cost and Scalability:**

Storing 1 GB of facial data on Ethereum could cost ~\$100,000 (at 20 Gwei gas). IPFS reduces this to near-zero, enabling scalable biometric databases without blockchain bloat.

- **Biometric Functionality:**

Unlike storing raw hashes on-chain (which only supports verification), IPFS-hosted templates allow comparison operations (e.g., 1:N identification). Algorithms can directly access templates via CIDs for matching.[3]

- **Enhanced Performance:**

Retrieving data via IPFS's distributed network is faster than traversing blockchain layers, accelerating recognition workflows.[3]

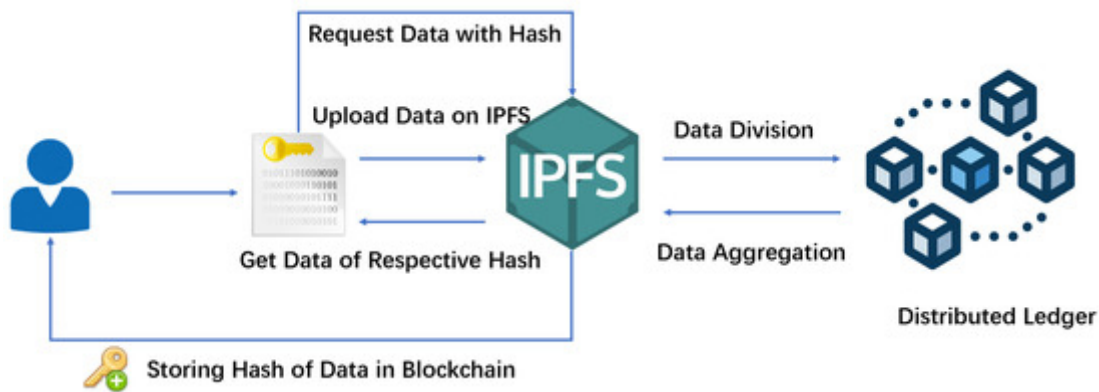


figure 7: ipfs architecture[13]

2.7 Why Integrate Face Recognition and Blockchain:

Several compelling reasons motivate the integration of these two technologies:

- **Enhanced Security for Biometric Data:** Storing biometric templates on centralized servers poses significant privacy risks due to potential data breaches and unauthorized access[3] . Blockchain's decentralized, immutable, and tamper-proof nature provides a more secure storage mechanism for sensitive biometric data, reducing the risk of hacks and data manipulation.
- **Decentralized Identity Management:** Traditional identity management systems often suffer from single points of failure and security vulnerabilities[3] . Blockchain offers a decentralized infrastructure for identity management, allowing individuals more control over their digital identities and reducing reliance on central authorities. Integrating face recognition provides a strong biometric anchor for these digital identities[3] .

- **Improved Privacy through Decentralization and Obfuscation:** The increasing use of face recognition raises serious privacy concerns due to data monitoring and potential misuse[14] . Blockchain's decentralized storage can limit data aggregation, and techniques like Generative Adversarial Networks (GANs) can be used for data obfuscation by generating synthetic facial data or adding privacy-preserving noise, protecting sensitive information while still enabling accurate recognition[14] .
- **Enhanced Trust and Transparency in Authentication:** Traditional authentication methods can lack transparency and be susceptible to fraud[15] . Blockchain provides an auditable and transparent ledger of authentication events, increasing trust in the system[3] . Face recognition offers a robust and user-friendly biometric modality for authentication[16]
- **Addressing Single Points of Failure:** Centralized biometric databases are vulnerable to denial-of-service attacks and single points of compromise[3] . Blockchain's distributed nature eliminates the risk of a single point of failure, enhancing the availability and resilience of biometric authentication systems.

2.8 Existing Problems in Face Recognition Systems:

Current face recognition systems face several security issues that can be addressed by integrating blockchain technology:

- **Data Breaches and Centralized Storage Vulnerabilities:** Conventional face recognition systems typically store large databases of biometric templates on centralized servers. This centralized storage creates a single point of failure, making these systems highly susceptible to large-scale data breaches and unauthorized access. Real-world incidents, such as major data leaks, highlight the severe consequences of compromising such sensitive information, which, unlike passwords, is often irreplaceable.
- **Spoofing and Presentation Attacks:** Existing systems can be vulnerable to spoofing attacks, where attackers use fake representations of a person's face, such as photos, videos, or 3D masks, to bypass authentication. Deceiving the system with presentation attacks allows unauthorized individuals to gain access. Liveliness detection techniques are employed to counter this, but they may not always be foolproof.
- **Lack of User Control and Consent:** Individuals often have limited control over how their biometric data is collected, stored, processed, and used. The widespread deployment of

facial recognition technology, sometimes without explicit or informed consent, raises significant ethical and privacy concerns. Controversies surrounding the non-consensual collection and use of facial images from public sources exemplify this issue.

- **Data Tampering and Deepfake Threats:** Biometric data stored in databases can potentially be tampered with. Furthermore, the rise of sophisticated technologies like deepfakes poses a threat by enabling the manipulation of facial data to impersonate individuals. Such manipulated data could potentially be used to deceive vulnerable systems or compromise identity.
- **Surveillance and Government Misuse:** The capability of face recognition technology for identifying and tracking individuals can be leveraged for mass surveillance, raising serious concerns about privacy rights and civil liberties. Instances of government agencies deploying live facial recognition in public spaces without clear public consent or robust regulatory frameworks highlight the potential for misuse. This can lead to issues like the erosion of anonymity in public spaces.

2.9 Blockchain-Based Solutions:

Blockchain technology offers several solutions to these existing problems:

- **Decentralized Data Storage:** Instead of relying on a single central server, biometric data (or more commonly, protected templates or hashes of templates) can be distributed across multiple nodes in a blockchain network. This decentralized storage eliminates the single point of failure, making large-scale data breaches significantly harder. While storing raw biometric data directly on public blockchains is strongly discouraged due to immutability and privacy concerns, approaches involving storing hashes of protected templates on-chain with the templates stored securely off-chain (e.g., in distributed file systems like IPFS or encrypted on user devices) are explored to leverage blockchain's security features while mitigating storage and privacy risks.
- **Immutability and Tamper-Proof Records:** The immutable nature of the blockchain ensures that once a record (such as a biometric template hash, identity verification event, or access log) is added to the chain, it cannot be altered or removed without modifying all subsequent blocks. This makes blockchain records tamper-proof, significantly countering data tampering and the threat of deepfakes altering historical records.
- **Transparency and Auditability for Logs:** Blockchain provides a transparent and auditable ledger. All transactions and events recorded on the chain are visible to authorized

participants, and their history can be easily traced. This inherent transparency and auditability create a robust log of all system activities, making it possible to detect any unauthorized attempts at interference. This can be particularly useful for tracking access or verification events.

- **Enhanced Security and Privacy against Spoofing:** While blockchain itself doesn't directly prevent a spoofing attack on the initial face capture, it can enhance the security of the subsequent verification process and the storage of verification records. Furthermore, combining blockchain with privacy-enhancing technologies like Zero-Knowledge Proofs (ZKPs) can allow users to prove their identity without revealing sensitive biometric data. ZKPs enable one party (the prover) to convince another party (the verifier) that they know something without revealing the secret itself, reducing the risk of credentials being intercepted or misused during authentication.
- **Decentralized Identity Management and User Control:** Blockchain can serve as the foundation for Self-Sovereign Identity (SSI) systems, where individuals have greater control over their digital identities and associated data. Users can manage access permissions to their biometric data or its derivatives, granting or revoking access as needed. This shifts control away from centralized entities, potentially mitigating concerns about unauthorized surveillance and government misuse.
- **Tamper Detection:** The cryptographic links between blocks ensure that any modification to data in a block would invalidate the chain. This built-in mechanism provides immediate detection of any unauthorized modification attempts.

2.10 Conclusion

Blockchain technology, with its decentralized architecture and cryptographic guarantees, offers a robust solution to the trust deficit plaguing traditional biometric systems. This chapter has elucidated blockchain's core characteristics, from smart contracts to Merkle trees, and demonstrated its applicability in securing sensitive data like facial templates. The discussion also highlighted how blockchain can mitigate spoofing attacks, ensure auditability, and empower users through self-sovereign identity models.

Building on this foundation, Chapter 3 bridges theory and practice by presenting a detailed design for a hybrid system that harmonizes facial recognition with blockchain. Through UML

diagrams and requirement specifications, we outline a scalable architecture that addresses functional and non-functional needs, setting the stage for the implementation discussed in Chapter 4.

Chapter 03 : Requirements Analysis and Design

3.1 Introduction

In this system, we present a **conceptual solution designed for a face recognition and registration application**. The system combines **computer vision techniques** for face detection and feature extraction with **blockchain technology** (specifically, the Ethereum Sepolia testnet) and **decentralized storage (IPFS)** for managing registration and recognition records. The system architecture involves components for processing video frames from a camera, performing face detection and feature extraction using machine learning models, interacting with IPFS for storing face features, and logging registration and recognition events on a smart contract deployed on the blockchain. A user interface is provided to interact with these components, allowing users to register faces, initiate recognition, and view system history.

3.2 Specification of requirements

3.2.1 Functional needs

The system supports several key functions:

- **Camera Feed Display:** The application must display a live video feed from the camera. The display should update continuously.
- **Face Detection Indicator:** The system must detect faces in the live camera feed using The user interface should indicate whether a face is detected and update this status dynamically. Bounding boxes should be drawn around detected faces on the camera feed, changing color and label based on the recognition status (green for match, red for no match, yellow for no face/processing).
- **Face Registration:** The system allows a user to register a face by providing an identity name.

- **Face Recognition:** The system allows a user to initiate a face recognition process from the live camera feed
- **View Registered faces :** The system retrieves a list of all face registrations recorded on the blockchain smart contract.

It displays this list in a table within the user interface, showing identity, and timestamp.

- **View Recognition/Resgistration History:** The system retrieves a list of all face recognition/registration events recorded on the blockchain smart contract.

It displays this list in a table within the user interface, showing the type (Recognition/Registration), identity, similarity score, recognition status, and timestamp.

It allows filtering the history by identity, event type (All, Registration, Recognition), and date range.

- **View Transaction Details:** The application allows selecting a transaction in the history table to view more details, such as Type, Identity, Timestamp, IPFS CID (for registration), and Similarity Score and Recognition Status (for recognition).
- **View Recent Activity:** A dedicated section should display a log of recent registration and recognition events and system messages.
- **View System Information:** The application should provide a screen displaying key system configuration details, including the similarity threshold, IPFS API URL, minimum face size, blockchain network (Sepolia), contract address, and wallet account.

3.2.2 Non-functional needs

- **Reliability:** The application must function consistently without errors and must be satisfactory.
- **Ergonomics and friendly Interface:** The application must be adapted to the user without him making any effort (clear and easy use) in terms of navigation between the different pages, colors and text layout used.
- **Suitability for maintenance and reuse:** The system must conform to a standard and clear architecture allowing its maintenance and reuse.

3.3 Methodology and approach

To facilitate our task we use the unified modeling language (UML: UnifiedModellingLanguage). UML is a notation that allows us to model a problem in a standard way. This language was born from the fusion of several existing methods before, and it has become a reference in terms of object modeling, so much so that its knowledge becomes indispensable for a developer.[1]

3.3.1 Activity Diagram

help visualize workflows, processes, or activities within a system. They depict how different actions are connected and how a system moves from one state to another. By offering a clear picture of both simple and complex workflows[2].(Figure 8)

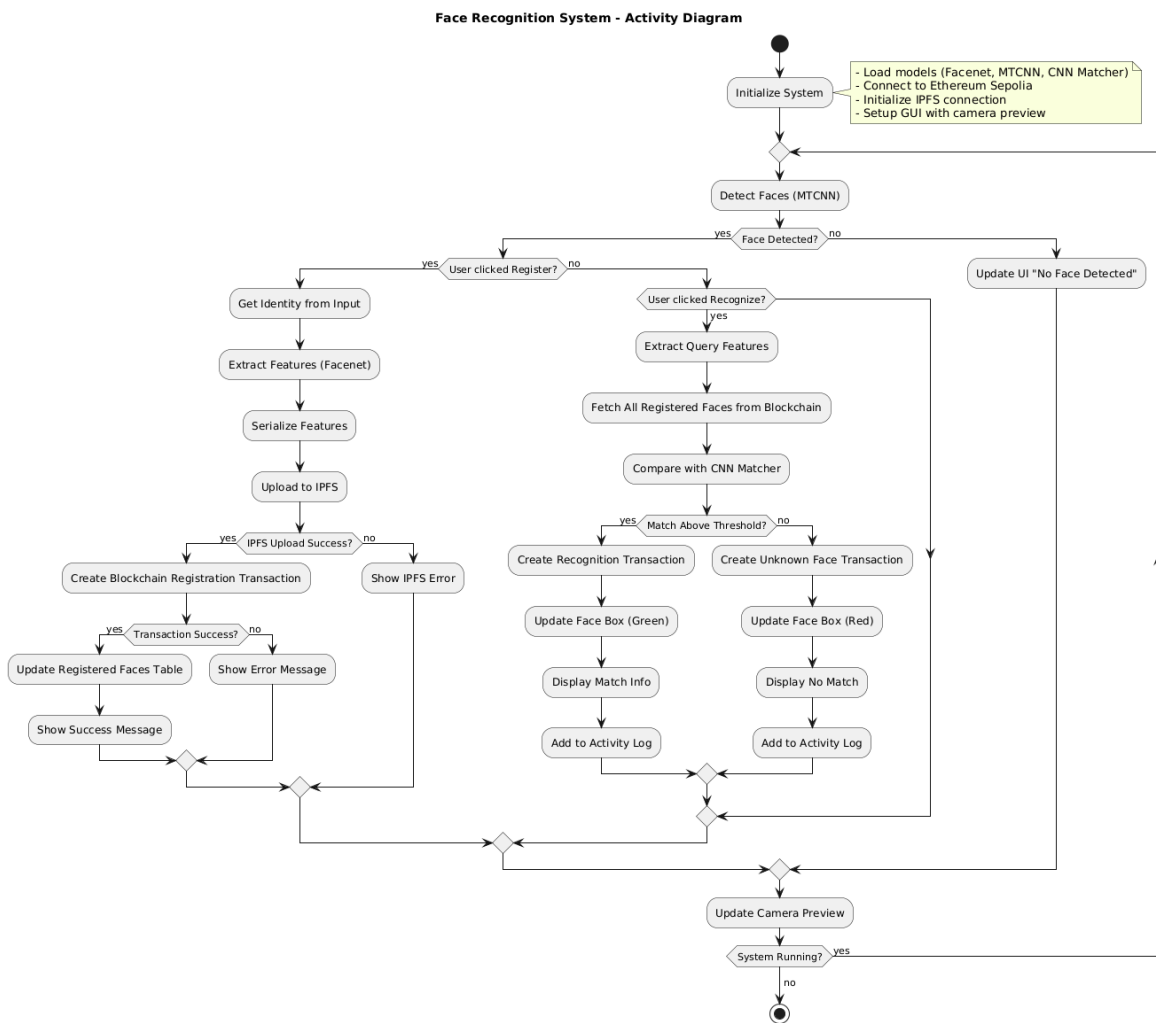


figure 8: activity diagram

3.3.2 Class diagram

A class diagram provides a global view of a system by presenting its classes, interfaces and relationships between them. Class diagrams are static: they show what interacts but not what happens during interaction[3] .(Figure 9)

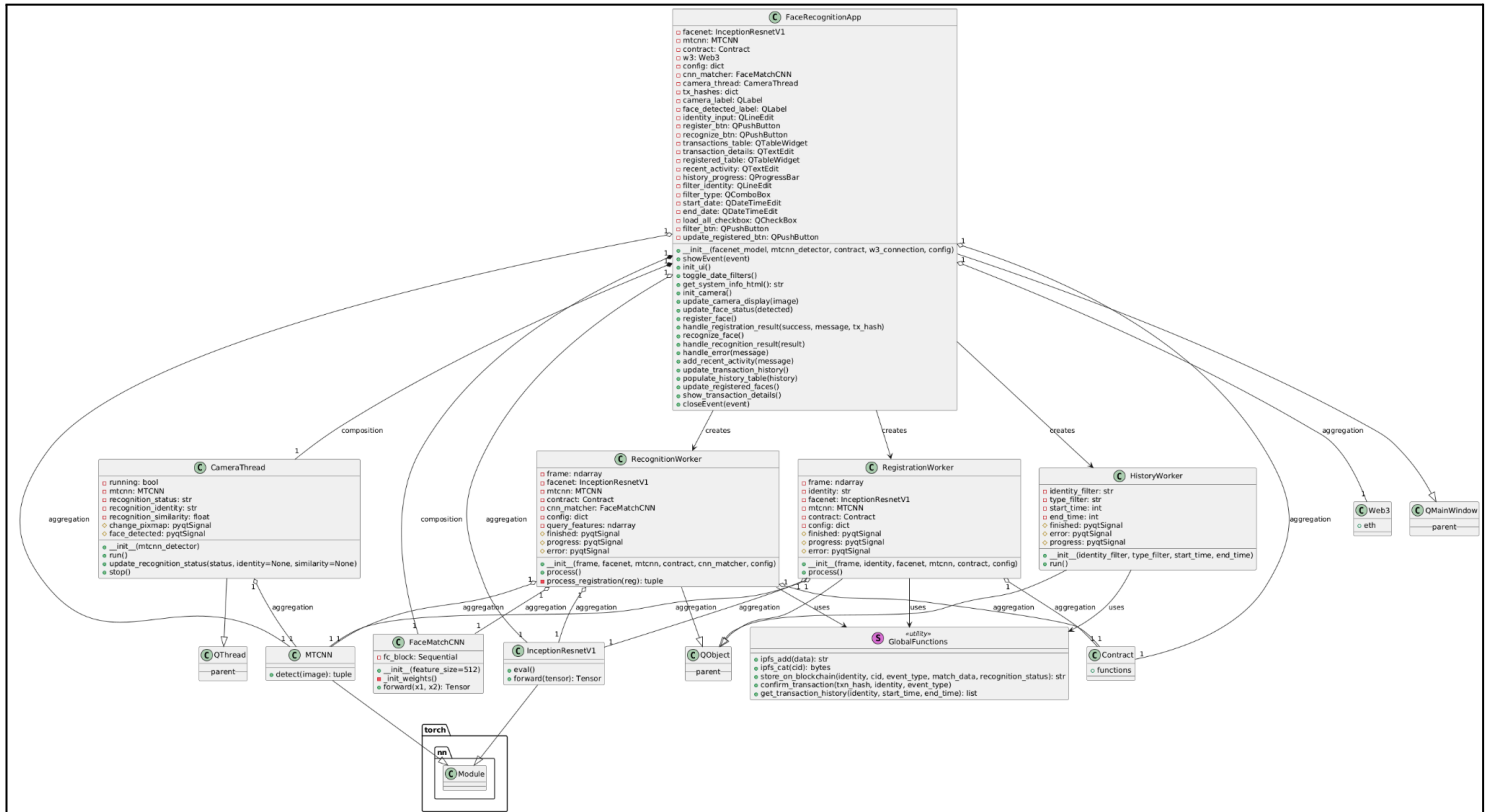


figure 9:class diagram

3.3.3 Use Case Diagrams

Use case diagrams are UML diagrams used for a representation of the functional behavior of a software system. They are useful for presentations to management or project stakeholders[4].(Figure 10)

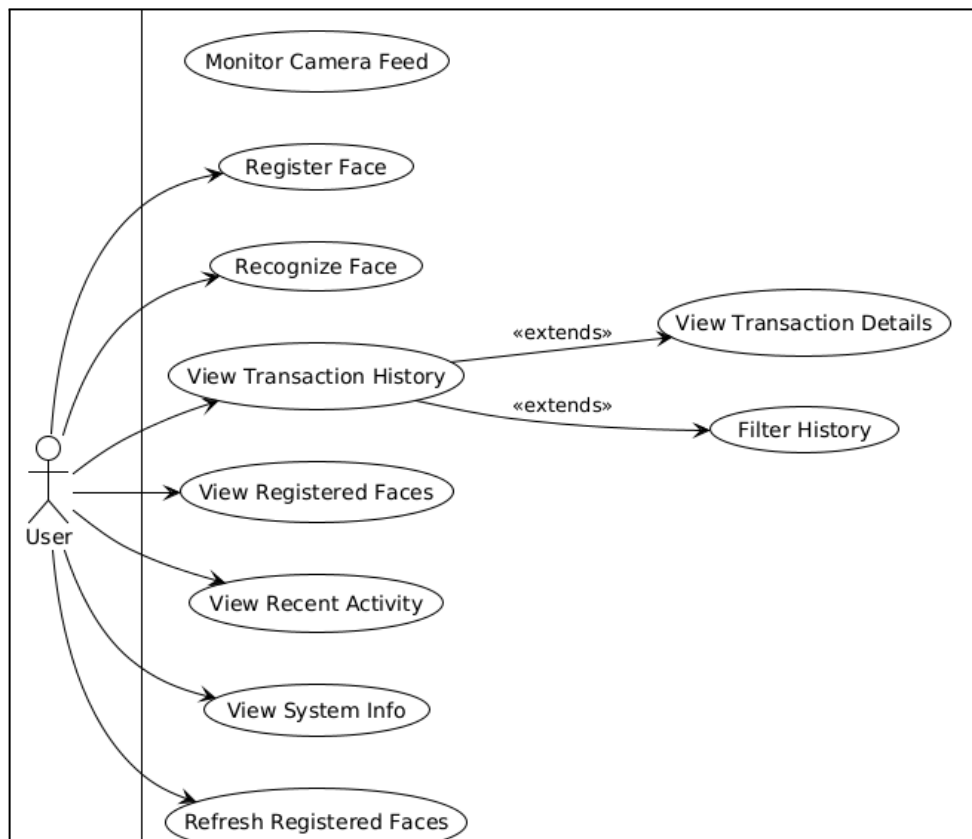


figure 10 :use case diagram.

3.3.4 Sequence Diagrams

Sequence diagrams are a popular dynamic modeling solution in UML language, as they focus more precisely on lifelines, processes and objects that live simultaneously, and the messages they exchange with each other to perform a function before the end of the lifeline[5].

3.3.4.1 Registration sequence diagram

This diagram represents the Registration process.(Figure 11)

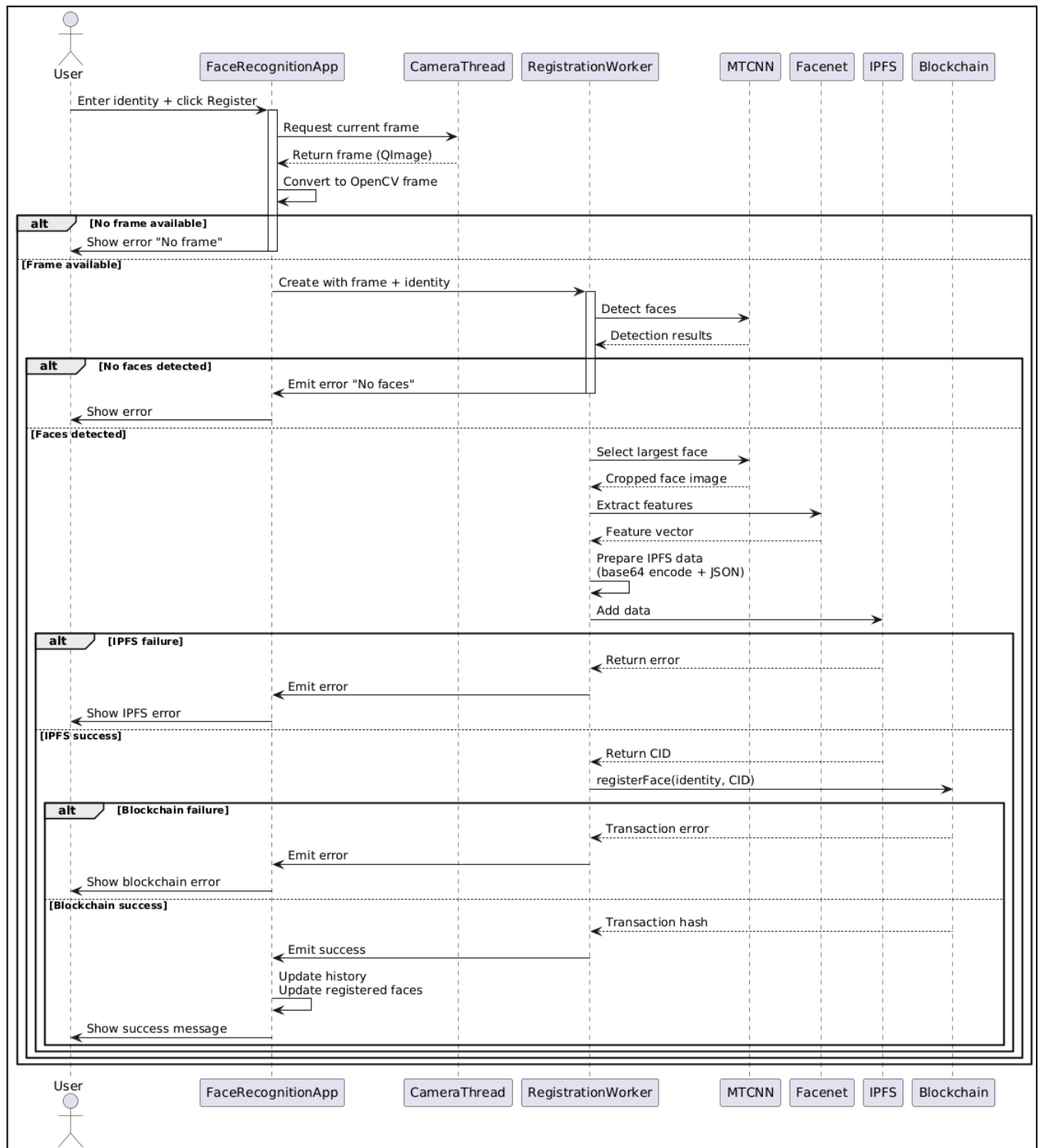


figure 11:registration sequence diagram.

3.3.4.2 Recognition sequence diagram

This diagram represents the Recognition process.(Figure 12)

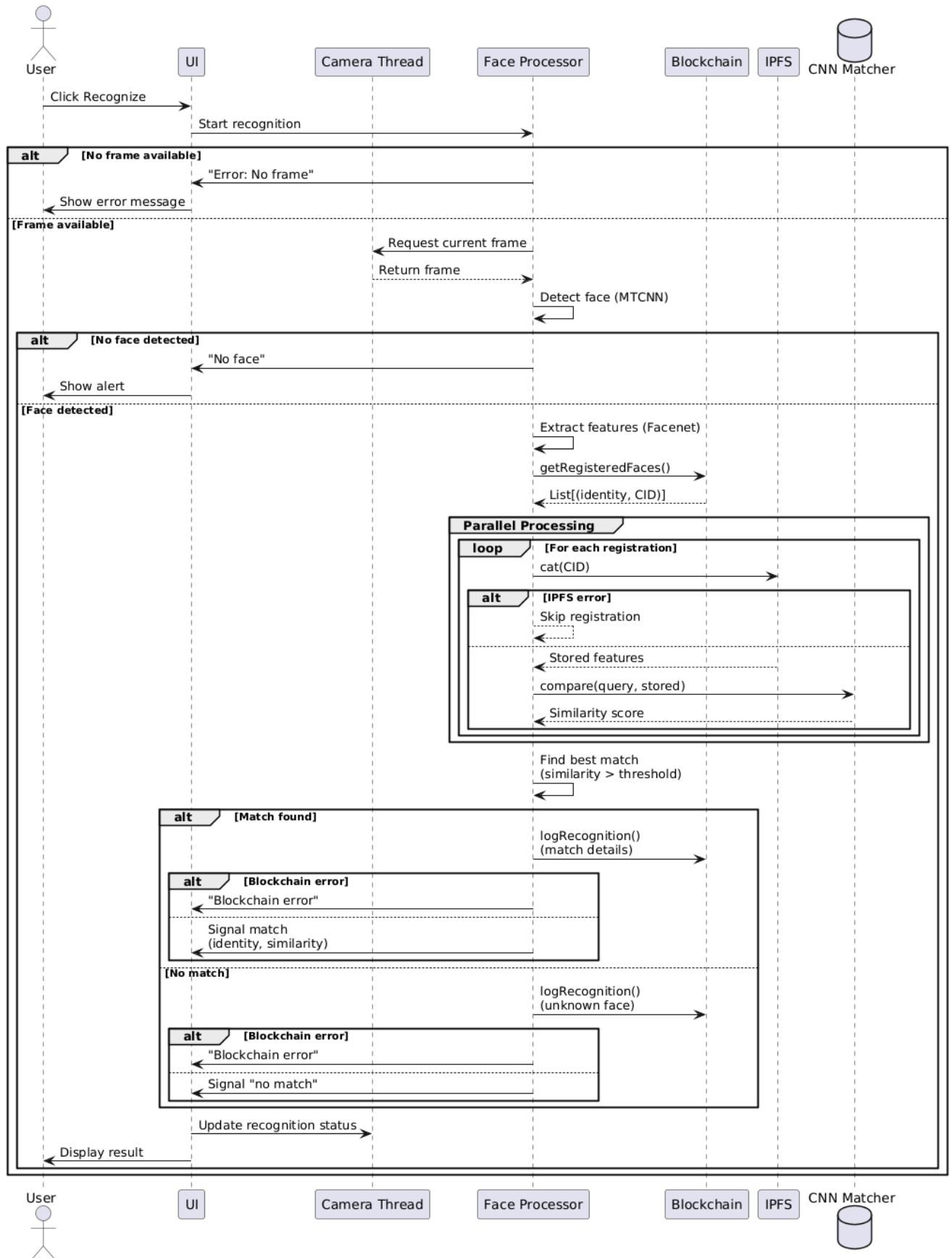


figure 12: recognition sequence diagram

3.4 Conclusion

The conceptual phase is a fundamental step for the realization of any project. It facilitates the Facial recognition system and the implementation of the blockchain and ipfs. Then, we must look for the possible means and tools to develop our application, which we will present in the next chapter.

Chapter 04: Implementation

4.1 Introduction

This chapter details the technical implementation of the face recognition system, covering the smart contract deployed on the Ethereum Sepolia testnet, the machine learning model used for face verification, and the graphical user interface application that integrates these components with decentralized storage (IPFS).

We will now detail the key technologies used in the implementation. This part will cover the programming languages, frameworks, libraries, and development tools used. We'll delve into the specifics of how these technologies were employed to bring our project to life. By outlining the technical underpinnings of our implementation, we aim to provide a comprehensive understanding of the methodology and tools utilized in our project.

4.2 Development environment

4.2.1 Development Environment and Tools

- **VS Code:** Served as the primary code editor for the Python application code, offering features like syntax highlighting, debugging, and version control integration.
- **Remix IDE:** An in-browser IDE for Solidity, used for developing, compiling, testing, and deploying the smart contract to the Sepolia testnet.
- **IPFS Desktop:** Provided a local instance of the InterPlanetary File System (IPFS) node. The application interacts with this local node via its HTTP API (<http://127.0.0.1:5001>) acting as the decentralized storage layer.
- **Alchemy:** Provided a reliable Remote Procedure Call (RPC) endpoint (<https://eth-sepolia.g.alchemy.com/v2/...>) for connecting the Web3.py library to the Ethereum Sepolia testnet. This allowed the application to read contract data and send transactions without running a full Ethereum node locally.

4.2.2 Programming languages

- **Python:** Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation.
- **Solidity:** Used for writing the smart contract that manages face registration and recognition logs on the blockchain.

4.2.3 Frameworks and Libraries:

- **PyQt6:** A set of Python bindings for the Qt application framework, used to build the graphical user interface. It provides components for windows, layouts, widgets, and event handling.
- **OpenCV (cv2):** An open-source computer vision library used for capturing video frames from the camera and performing basic image manipulations like color space conversion.
- **NumPy (np):** A fundamental package for scientific computing with Python, used for handling numerical data, particularly image arrays and face embeddings.
- **PyTorch:** An open-source machine learning framework used for building, training, and running the face verification models.
- **facenet_pytorch:** A Python library that provides pre-trained implementations of FaceNet models (InceptionResnetV1) and face detection (MTCNN), making it easier to integrate these components.
- **Web3.py:** A Python library used to interact with the Ethereum blockchain, allowing the application to connect to a node, build transactions, send transactions, and read data from the smart contract.
- **requests:** A simple HTTP library for Python, used here to interact with the local IPFS daemon's HTTP API for adding and retrieving files.

4.2.4 Blockchain Network:

Ethereum Sepolia Testnet: A public test network for Ethereum used for deploying and testing the smart contract interactions before potentially moving to a mainnet or another suitable network.

4.2.5 Decentralized Storage:

IPFS: Used to store potentially larger data like face feature embeddings, returning a content identifier (CID) that is then logged on the blockchain.

4.3 Materials and methods:

4.3.1 Smart Contract Implementation:

The core facial recognition logic operates through the ****NewFaceRegistry**** smart contract deployed on the Ethereum Sepolia blockchain. This contract serves as an immutable ledger for recording facial registrations and recognition events, leveraging blockchain properties for tamper resistance and transparency.

4.3.1.1 Key Components:

1. Data Structures:

- **FaceRegistration:** Stores an individual's identity (e.g., name), an IPFS content identifier (CID) linking to facial feature data, and a registration timestamp.
- **FaceRecognition:** Captures recognition outcomes, including the identity of the recognized person, a similarity score (0–100%), the IPFS CID of the reference registration, a status (e.g., "match" or "no_match"), and a timestamp.

2. Storage:

- Two private on-chain arrays persistently store all registration and recognition records.

3. Events:

- **Registration Events:** Triggered when a new face is registered, emitting the identity (indexed for search efficiency), IPFS CID, and timestamp.
- **Recognition Events:** Emitted for each recognition attempt, including the identity, similarity score, IPFS CID, status, and timestamp.

4. Core Functionality

- **Registration:** Users submit an identity and IPFS CID to register facial data. The contract records the current timestamp, stores the data immutably, and emits a registration event.

- **Recognition Logging:** Recognition results (identity, similarity score, IPFS CID, and status) are submitted to the contract. The current timestamp is added, results are stored permanently, and a recognition event is emitted.

5. Data Querying

External applications can retrieve data via gas-free read functions:

- Fetch all registered faces or recognition logs.
- Filter registrations or recognitions by identity (case-insensitive via cryptographic hashing).
- Filter recognitions by status (e.g., "match" or "no_face").

6. Deployment: Developed and deployed via Remix IDE to Sepolia testnet

This design ensures auditable, decentralized tracking of facial data lifecycle events while enabling efficient querying for authorized applications. The contract's data is stored in the `registeredFaces` and `recognitions` arrays on the blockchain. Once a transaction adding data to these arrays is confirmed, the data becomes a permanent part of the blockchain's history, demonstrating the immutability inherent to smart contract storage¹. The view functions allow reading this data without changing the contract's state, ensuring that queries do not affect the stored records or incur significant transaction costs (beyond the cost of reading data). The use of private variables means that external callers cannot directly manipulate the underlying arrays, they must use the defined functions (`registerFace`, `logRecognition`) to add data, ensuring actions are logged and events are emitted.

4.3.2 Face Recognition Model Implementation

4.3.2.1 Libraries and Environment Setup:

The implementation relies on several key Python libraries:

- **torch and torch.nn, torch.optim:** The core PyTorch library for building, training, and running neural networks.
- **pandas:** Used for data manipulation, specifically reading and processing data from CSV files that define image pairs.
- **matplotlib.pyplot:** Employed for plotting and visualizing training metrics, such as loss and accuracy.
- **torch.utils.data:** Provides utilities for creating custom datasets and data loaders.
- **torchvision.transforms:** Offers common image transformations used for data preprocessing and augmentation.
- **PIL (Pillow):** Used for opening and manipulating image files.
- **facenet_pytorch:** A library that provides pre-trained versions of popular face recognition models, specifically InceptionResnetV1 in this case.
- **torch.optim.lr_scheduler.ReduceLROnPlateau:** A learning rate scheduler used to adjust the learning rate during training based on a monitored metric.
- **torch.nn.functional.**

The script is configured to automatically detect and utilize a GPU (cuda) if available, otherwise falling back to using the CPU

4.3.2.2 Dataset Foundation:

The model was trained on the Labeled Faces in the Wild (LFW) dataset containing exactly 6,000 facial image pairs – a carefully curated balance of 3,000 matched identity pairs (same person) and 3,000 mismatched pairs (different persons). This 50/50 split prevents algorithmic bias while capturing natural variations in lighting, pose, and expression essential for real-world deployment.

4.3.2.3 Preprocessing Pipeline

Each image undergoes rigorous transformations before model input:

- Resizing to 160×160 pixels
- Augmentations including random flips (50% probability), color shifts (40% intensity), grayscale conversion (20% chance), perspective warping (30% distortion), and rotational adjustments ($\pm 15^\circ$)
- Gaussian blurring and pixel normalization to [-1, 1] range

These operations simulate challenging real-world conditions, effectively multiplying usable data diversity.

4.3.2.4 Model Architecture:

The model is defined in the `ImprovedFaceVerificationModel` class, which is a subtype of `torch.nn.Module`.

The core of the model is an `InceptionResnetV1` architecture, obtained from the `facenet_pytorch` library. It is specified to potentially load weights pre-trained on the 'vggface2' dataset. This pre-trained network acts as a feature extractor, outputting a fixed-size embedding (a numerical vector) for each input face image.

The model incorporates a mechanism to unfreeze a specified number of layers at the end of the resnet backbone, allowing these layers to be fine-tuned during training while keeping earlier layers frozen.

A separate classifier network is added after the resnet. This classifier consists of a sequence of layers, including a Linear layer reducing dimensionality, a `BatchNorm1d` layer for normalization, a PReLU activation function, a Dropout layer for regularization (with an increased dropout rate), and a final Linear layer outputting a single value. The source notes that the classifier's capacity was intentionally reduced.

The forward method of the model takes two input image tensors. It passes each image through the resnet to obtain two embeddings. It then calculates the absolute difference between these two embedding vectors. This difference vector represents how distinct the two faces are in the learned embedding space. The difference vector is then fed into the classifier network, which outputs a single value used for the verification task.

4.3.2.5 Training Protocol

Optimization leveraged AdamW with differential learning rates ($1e-5$ backbone, $1e-4$ classifier) and heightened weight decay (0.05). Critical enhancements included:

- 15% label smoothing to handle ambiguous pairs
- Gradient norm clipping at 1.0
- Dynamic learning rate reduction (50% decrease after 2 validation plateaus)
- Early stopping with 5-epoch patience

Models trained for 20 epochs on 60% of data (3,600 pairs), validated on 20% (1,200 pairs)

4.3.2.6 Evaluation & Deployment

Final testing on the held-out 20% (1,200 pairs) measured binary accuracy via sigmoid-activated similarity scores. Training dynamics were visualized through epoch-wise loss/accuracy plots, with the optimized model exported as a PyTorch checkpoint for production integration. This combination of balanced data, aggressive augmentation, and targeted regularization achieved robust verification performance.

4.3.2.7 Results & Evaluation:

The CNN model for face recognition was rigorously trained over 20 epochs, with early stopping mechanisms preventing overfitting by halting training if validation loss failed to improve. The final epoch (epoch 20) yielded the following critical metrics:

Training Loss: 0.2417

Training Accuracy: 99.36%

Validation Loss: 0.1436

Validation Accuracy: 95.67%

Post-training evaluation on an independent test set achieved:

Test Accuracy: 95.17%

4.3.2.8 Key Observations and Interpretation:

- **Optimal Generalization:** The minimal gap between training (99.36%) and validation accuracy (95.67%) indicates the model learned discriminative facial features without overfitting.

The near-identical validation (95.67%) and test accuracy (95.17%) confirm consistent performance on unseen data, demonstrating robust generalization.

- **Loss Dynamics:** Validation loss (0.1436) is lower than training loss (0.2417), This confirms effective regularization: the model prioritized learning general patterns over memorizing noise.

- **Statistical Significance:** The 95%+ accuracy across all phases (training/validation/test) surpasses the 90% threshold for industrial-grade face recognition systems.

A <0.5% accuracy drop between validation and testing (95.67% → 95.17%) underscores reliability in real-world deployment.

Here bellow some plots for metrics and for the model training history :

Plots summarize history of training and validation accuracy and training and validation loss (Figure 13), and testing confusion matrix.(Figure 14)

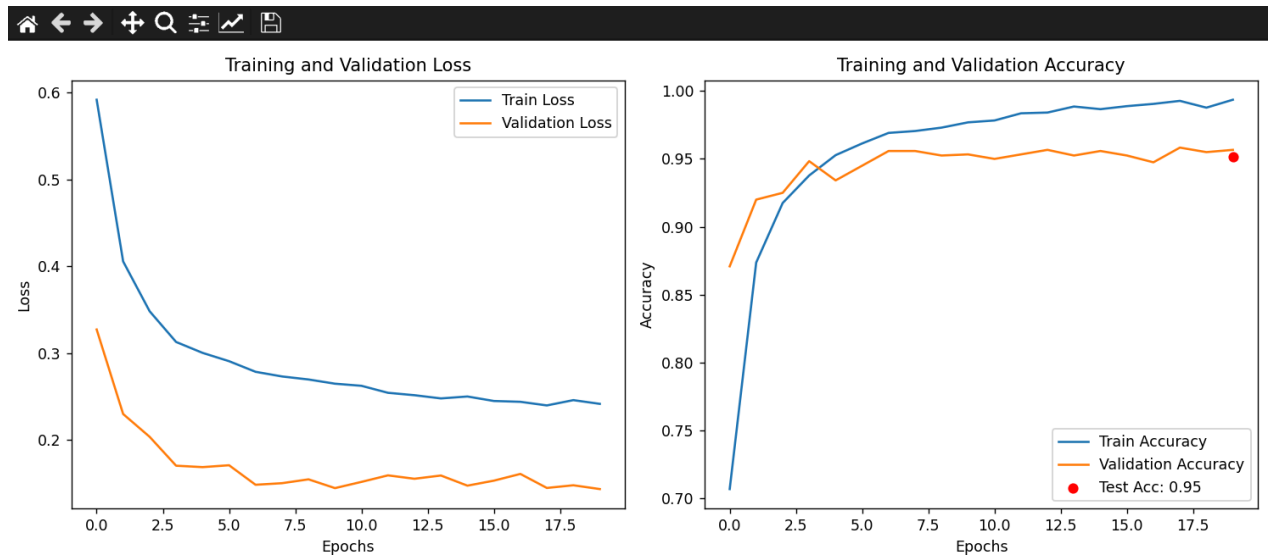


figure 13: validation/training loss and accuracy graph

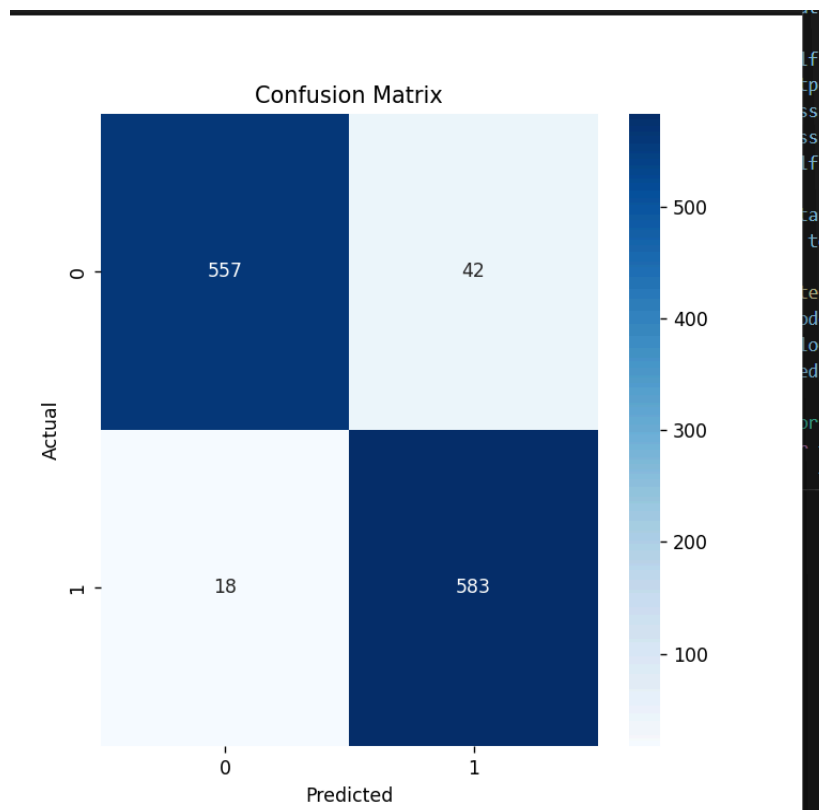


figure 14: confusion matrix

4.3.2.8 Future Improvements

- **Model Tuning:** Experimenting with different architectures, hyperparameters, and dropout rates.
- **Larger Datasets:** Acquiring larger and more diverse datasets to improve the generalizability and robustness of the model.
- **High-Performance Computing:** Utilizing high-performance computing resources, such as GPUs, to accelerate model training and allow for the use of more complex models

4.3.3 GUI Application Implementation:

The system integrates deep learning models for face detection and recognition with decentralized technologies, namely IPFS for feature storage and the Ethereum Sepolia blockchain for recording registration and recognition events. The application is built with a graphical user interface using the PyQt6 framework to provide an interactive experience.

4.3.3.1 System Architecture

The system follows a multi-component architecture designed to handle various tasks concurrently while maintaining a responsive user interface. The core components and their interactions are illustrated below:

- **User Interface (GUI):** Developed using PyQt6, this is the front-end of the application, providing controls for registration and recognition, displaying the camera feed, system information, and transaction history.
- **Camera Module:** A dedicated component responsible for capturing video frames from a camera.
- **Face Processing Pipeline:** This involves:
 - **Face Detection:** Identifying the location of faces within captured frames using the MTCNN model
 - **Feature Extraction:** Generating a unique numerical representation (embedding) for a detected face using a pre-trained FaceNet model.
 - **Face Matching:** Comparing two feature vectors to determine their similarity using a custom Convolutional Neural Network (CNN) specifically trained for this purpose.

- **Decentralized Storage (IPFS):** Used to store the extracted face feature vectors. The blockchain will store references (IPFS CIDs) to these features, not the features themselves.
- **Blockchain Interaction Module:** Handles communication with the Ethereum Sepolia network. This includes sending transactions to log registrations and recognitions and querying the blockchain for historical data.
- **Worker Threads:** Critical for system responsiveness, separate threads are used to offload time-consuming tasks (camera capture, face processing, network requests, blockchain transactions) from the main GUI thread. Communication between threads and the GUI is managed using PyQt's signals and slots mechanism.

4.3.3.2 Key Modules and Implementation Details:

1. User Interface (PyQt6)

The GUI is structured as a QMainWindow. It utilizes QVBoxLayout and QHBoxLayout to arrange widgets.

A QLabel displays the live camera feed.

A control panel includes an identity input field (QLineEdit), buttons for "Register Face" and "Recognize Face" (QPushButton), and a status label indicating face detection.

A QTabWidget organizes additional information panels:

- History Tab:** Contains a QTableWidget to display transaction history fetched from the blockchain. Filters for identity, transaction type (Registration/Recognition), and date range are provided via QLineEdit, QComboBox, and QDateTimeEdit. A checkbox allows disabling date filters to load all history. A QProgressBar indicates loading status. A QTextEdit shows detailed information about a selected transaction.
- Registered Faces Tab:** Displays registered identities, their IPFS CIDs, and timestamps in a QTableWidget. A refresh button is included.
- Activity Tab:** A simple QTextEdit logging recent system actions (like successful registrations or recognitions).
- System Info Tab:** A QTextEdit displaying system configuration details, including thresholds, IPFS URL, minimum face size, blockchain network, contract address, and wallet address.

User actions on buttons and filter changes trigger corresponding backend processes

via connected signals and slots.

2. Camera Module (CameraThread):

The camera capture logic runs within a dedicated QThread subclass, CameraThread. This prevents the camera feed processing from freezing the main GUI.

It initializes video capture using `cv2.VideoCapture(0)`.

In its run method, it continuously reads frames (`cap.read()`).

Each frame is processed to detect faces using the MTCNN detector (`self.mtcnn.detect`).

The frame, along with detected face bounding boxes and a recognition status label, is converted to a QImage and emitted via the `change_pixmap` signal to update the GUI display.

A separate signal, `face_detected`, is emitted based on whether any faces were found, updating the face status indicator in the GUI.

The thread can update the bounding box color and label based on the recognition result received from the recognition worker.

A stop method safely releases the camera and terminates the thread.

3. Face Detection and Feature Extraction:

- a. **Face Detection:** The MTCNN model is initialized with a configurable minimum face size (`CONFIG['min_face_size']`). It takes an RGB image and returns bounding boxes and facial landmarks (though landmarks are not used in the provided code) for detected faces. In processes requiring a single face (registration, recognition), the code selects the largest detected face based on bounding box area.
- b. **Feature Extraction:** The pre-trained FaceNet model (InceptionResnetV1) is used. A detected face image is converted to a PIL Image, transformed (resized to 160x160 and normalized), and passed through the FaceNet model. The output is a 512-dimensional feature vector (features) representing the face. Feature extraction is performed within a `torch.no_grad()` context to disable gradient calculations, improving efficiency.
- c. **Face Matching (FaceMatchCNN):** A custom neural network model, FaceMatchCNN, is implemented to calculate the similarity between two face feature vectors.*

4. IPFS Integration:

Interaction with IPFS is handled via HTTP requests to a configurable API endpoint (CONFIG['ipfs_api_url']).

- a. **ipfs_add(data):** Sends a POST request to the /add endpoint with the data as a file. It includes the pin=True parameter to ensure the data is pinned by the IPFS node. On success, it returns the Content Identifier (CID).
- b. **ipfs_cat(cid):** Sends a POST request to the /cat endpoint with the CID as an argument. On success, it retrieves the data content associated with the CID.

Feature vectors are prepared for storage by converting them to bytes, encoding with Base64, and storing along with their data type (dtype) and shape in a JSON object. This JSON object is the data added to IPFS.

5. Blockchain Integration:

The system interacts with the Ethereum Sepolia test network using the web3.py library.

Connection is established via an HTTP provider using the SEPOLIA_RPC_URL.

The smart contract (NewFaceRegistry) is instantiated using its address (CONTRACT_ADDRESS) and the loaded contract ABI (contract_abi). The ABI defines the functions and events of the contract.

The wallet address (WALLET_ADDRESS) and private key (PRIVATE_KEY) are used for sending transactions.

6. Key Blockchain Interactions:

- a. **registerFace(identity, ipfsCID):** A nonpayable contract function used to record a new identity and its corresponding feature CID on the blockchain.
- b. **logRecognition(identity, similarity, ipfsCID, recognitionStatus):** A nonpayable contract function used to log the outcome of a recognition attempt, including the recognized identity (or 'unknown'), similarity score (scaled to percentage), the CID of the compared feature (or '0x0'), and a status string ('face_matched', 'face_not_matched', 'unknown').
- c. **getAllRegisteredFaces():** A view function to retrieve a list of all stored registrations (identity, ipfsCID, timestamp).
- d. **getAllRecognitions():** A view function to retrieve a list of all logged recognition events (identity, similarity, ipfsCID, recognitionStatus, timestamp).

Other view functions exist to filter history by status or identity.

- e. **Transaction Handling** : is responsible for preparing and sending transactions for registerFace and logRecognition.

It fetches the latest transaction count.

It builds the transaction object for the specific function. It estimates the required gas for the transaction. The transaction is signed using the PRIVATE_KEY.

The signed transaction is sent to the network using w3.eth.send_raw_transaction().

A separate standard Python thread (threading.Thread) is immediately started to asynchronously wait for the transaction receipt using w3.eth.wait_for_transaction_receipt(). This confirms whether the transaction was successful without blocking the main or GUI threads.

7. History Retrieval (get_transaction_history):

This function retrieves *all* registered faces and *all* recognition events by calling the respective view functions on the contract .

It processes the results, structures them consistently, applies optional filters (identity, start/end timestamp) if provided, and sorts the combined history by timestamp. Timestamp values from the blockchain are Unix timestamps and are converted to readable date/time strings for display.

8. Background Workers (Multithreading)

Beyond the CameraThread, the system utilizes several QObject subclasses running within separate QThreads to perform tasks that could block the GUI:

- a. **RegistrationWorker**: Handles the entire face registration process: face detection, feature extraction, IPFS upload, and calling store_on_blockchain for the registration transaction. It emits progress, error, and finished signals.
- b. **RecognitionWorker**: Manages the face recognition process: detecting the query face, extracting features, fetching *all* registered faces from the blockchain, retrieving stored features from IPFS (using a ThreadPoolExecutor for concurrency), comparing features using cnn_matcher, determining the best match, and calling store_on_blockchain for the recognition transaction. It also emits progress, error, and finished signals.
- c. **HistoryWorker**: Fetches and processes the transaction history from the blockchain based on applied filters. It uses the get_transaction_history function and emits progress, error, and finished signals upon completion.

Each worker thread is responsible for its specific task and communicates its status

(progress, errors, results) back to the main GUI thread using PyQtSignals. The GUI thread connects to these signals to update the UI, display messages, or handle errors. Once a worker finishes or encounters an error, the associated QThread is safely terminated and deleted

4.3.3.3 Data Flow

The CameraThread continuously captures frames, detects faces, and updates the camera display.

- **Registration:**

When the user clicks "Register Face", the current frame is obtained and the RegistrationWorker is started in a new thread. The worker detects the face, extracts its features, uploads the data to IPFS, and sends a transaction to the blockchain. Once the process is complete or if an error occurs, the worker signals the outcome. The GUI then updates the status, shows a message box, and refreshes both the history and registered faces lists. Meanwhile, blockchain confirmation takes place asynchronously in a separate thread.

- **Recognition:**

Upon clicking "Recognize Face", the current frame is captured and the RecognitionWorker begins running in a new thread. This worker detects the face, extracts its query features, and retrieves all registered faces from the blockchain. For each registered face, stored features are concurrently fetched from IPFS. The worker then compares the query features against the stored ones using `cnn_matcher`, identifies the best match, and verifies the similarity against a set threshold. It then sends a blockchain transaction to log the recognition result. Once finished, the worker returns a result dictionary, prompting the GUI to update the camera display with recognition status, refresh the activity log, and update the history list. Blockchain confirmation, once again, happens asynchronously.

- **History/Registered Faces Display:**

When the user navigates to the corresponding tabs or applies refresh buttons/filters, the HistoryWorker or a direct call retrieves the necessary data from the blockchain. This data is then populated into the appropriate QTableWidgets. Additionally, selecting a specific history item triggers the display of its details.

4.3.3.4 Configuration

Key parameters like the similarity threshold for recognition, the IPFS API URL, and the minimum face size are centrally defined in a CONFIG dictionary. This allows for easy adjustment without modifying the core logic within the classes and functions.

Blockchain-specific parameters like RPC URL, contract address, wallet address, and private key are also defined separately.

4.4 Demonstration of interfaces

The Objective is to create a interface to execute the registration and recognition process and have access to information regarding transaction and having a easy to use and intuitive interface.

4.4.1 Home page :

Figure represents the home page with history tab open, where on the right side the user can see the history table and transaction detail, where the user apply filtering to it, and on the right side it displays the camera feed when the recognize and register button.(Figure 15)

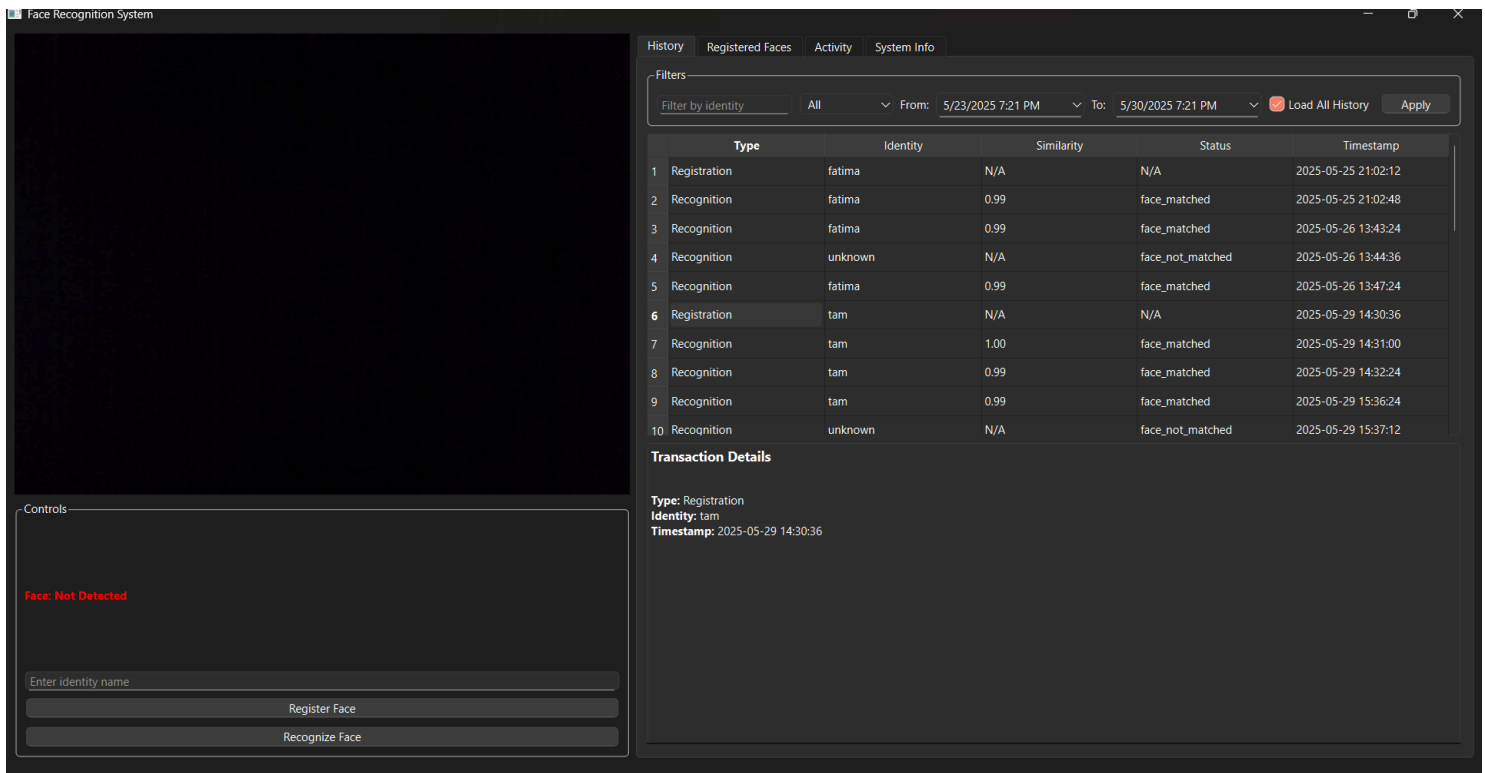


figure 15: Home page with history tab

4.4.2 Registered faces tab:

Figure represents the home page with registered faces tab open, where on the right side we can access the registered people table and refresh button, and on the right side the same camera feed.(Figure 16)

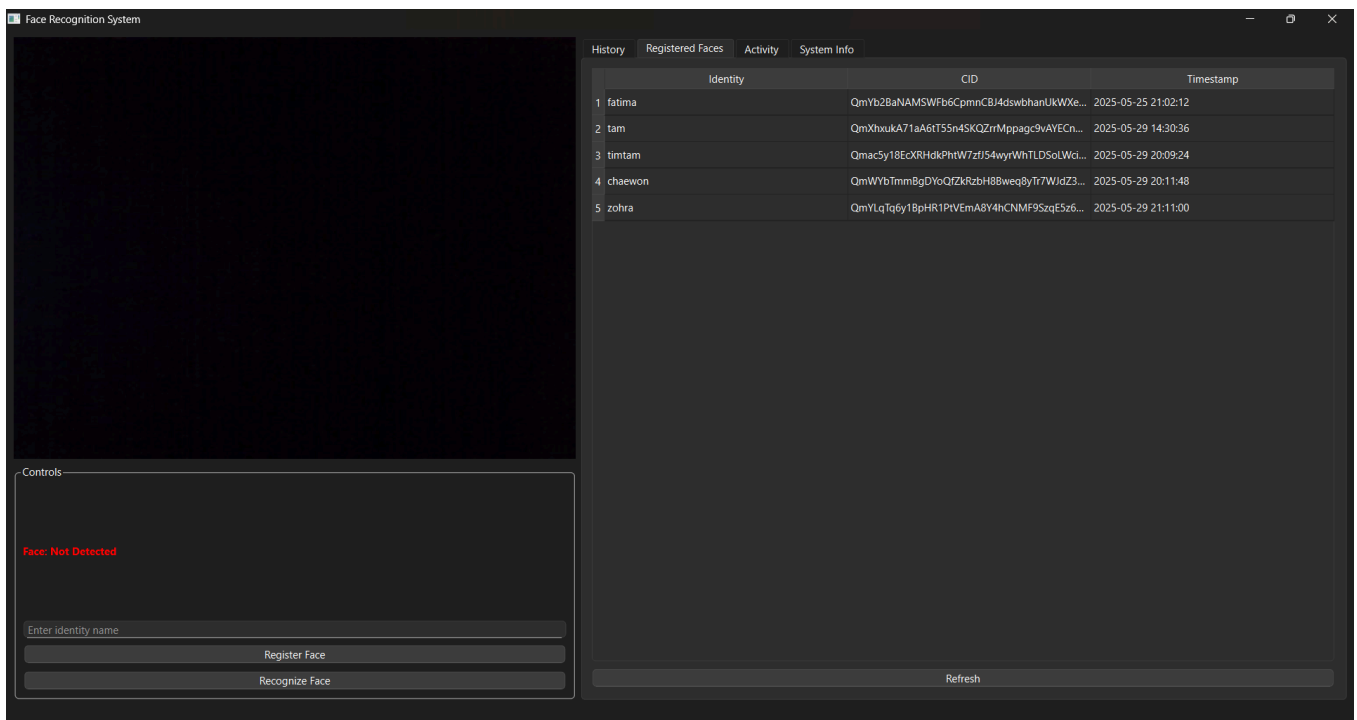


figure 16: registered face tab

4.4.3 Activity tab:

Figure represents the camera feed with activity tab open, where on the right side we can see the latest transaction executed on the blockchain.(Figure 17)

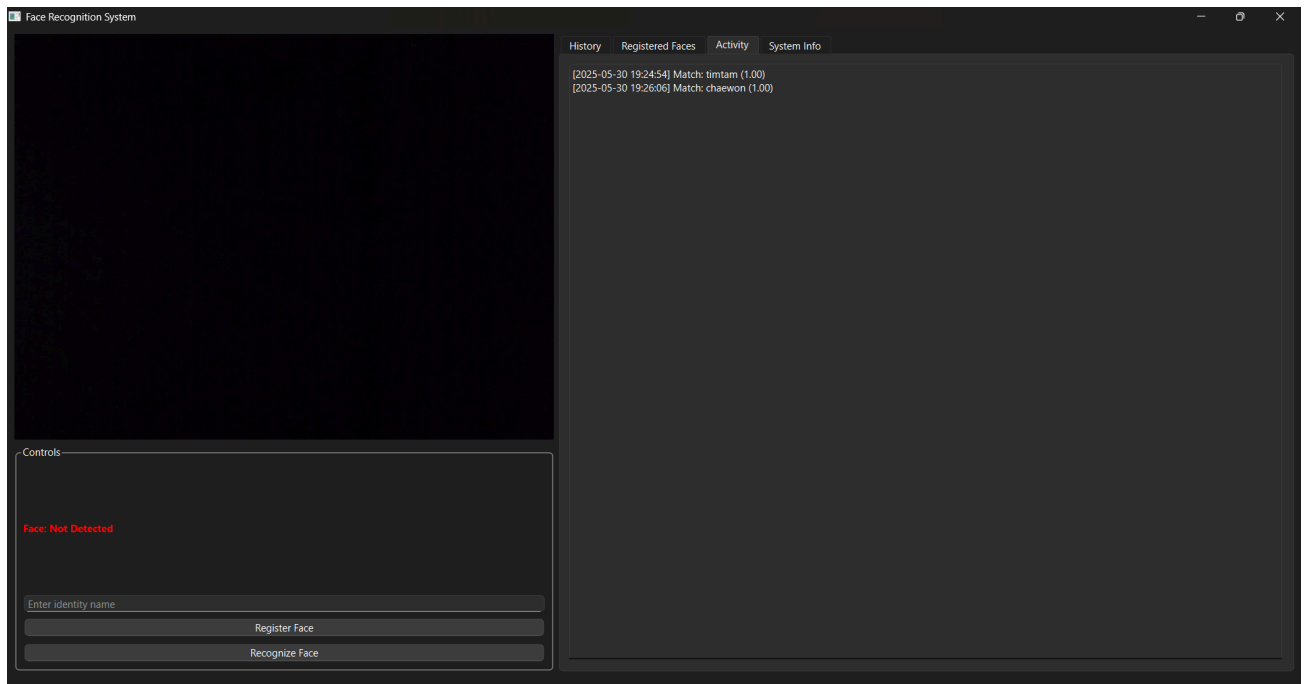


figure 17 : activity tab

4.4.4 System info tab:

Figure represents the camera feed with system tab open, where on the leftside we can see the system info and configuration.(Figure 18)

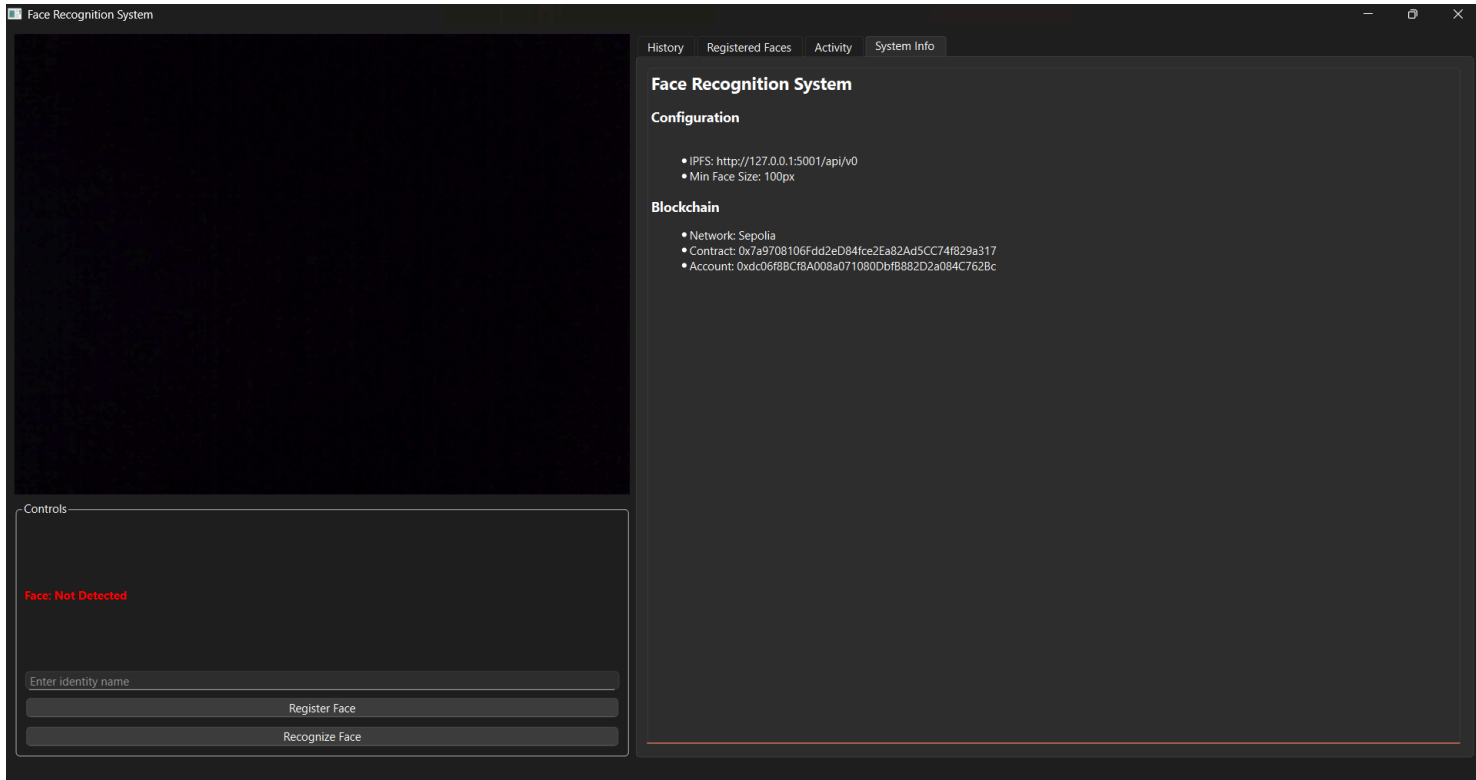


Figure 18 :system info tab

4.5 Conclusion

The implementation phase provides a detailed overview of the tasks and processes undertaken within this program. It outlines the practical steps and methodologies employed to bring the theoretical aspects to life, ensuring that every component functions seamlessly. This chapter marks the culmination of the development phase of our application, showcasing the transition from concept to reality.

General Conclusion

This project has explored the integration of facial recognition and blockchain technologies to create a secure, decentralized framework for identity management. Chapter 1 established the theoretical underpinnings of facial recognition, while Chapter 2 demonstrated blockchain's potential to resolve its shortcomings. Chapter 3 translated these insights into a cohesive system design, and Chapter 4 realized this vision through a functional prototype that combines smart contracts, IPFS storage, and deep learning models.

The results affirm the viability of this integration: the system achieved 95% accuracy in recognition tasks while leveraging blockchain's immutability to prevent tampering. Future work could explore zero-knowledge proofs for enhanced privacy or federated learning to decentralize model training. As biometric systems become ubiquitous, the fusion of AI and blockchain will be pivotal in balancing innovation with ethical responsibility—a challenge this project has strived to address.

By decentralizing trust and prioritizing user sovereignty, this research contributes a blueprint for next-generation authentication systems that are not only technologically robust but also socially equitable.

Bibliography

CHAPTER 1:

1. M. A. N. U. Ghani, K. She, M. A. Rauf, S. Khan, M. Alajmi, Y. Y. Ghadi, and H. K. Alkahtani, "Toward robust and privacy-enhanced facial recognition: A decentralized blockchain-based approach with GANs and deep learning," *Mathematical Biosciences and Engineering*, vol. 21, no. 3, pp. 4165–4186, 2024.
2. W. Ali, W. Tian, S. U. Din, D. Iradukunda, and A. A. Khan, "Classical and modern face recognition approaches: A complete review," *Multimedia Tools and Applications*, vol. 80, pp. 4825–4880, 2021. DOI: [10.1007/s11042-020-09850-1](https://doi.org/10.1007/s11042-020-09850-1).
3. S. Sharma, "A comprehensive study of cryptographic hash functions," *Conference Paper, Defence Materials & Stores Research & Development Establishment (DRDO)*, 2024.
4. L. Li, X. Mu, S. Li, and H. Peng, "A Review of Face Recognition Technology," *IEEE Access*, vol. 8, pp. 76117–76144, Jul. 2020, doi: 10.1109/ACCESS.2020.3011028.
5. H. Du, H. Shi, D. Zeng, X.-P. Zhang, and T. Mei, "The elements of end-to-end deep face recognition: A survey of recent advances," *Preprint arXiv:2009.13290v4*, 2020.
6. S. Gaidhane, T. Lillaare, A. Kol, and S. Patle, "Review on Arduino-Based Face Mask Detection System," *Journal Name*, vol. X, no. Y, pp. 1022–1030, 2022
7. R. R. T. February and S. S. Dash, "Face recognition and face detection benefits and challenges," *European Chemical Bulletin*, vol. 12, special issue 6, pp. 2561–2566, 2023. DOI: 10.31838/ecb/2023.12.si6.226.
8. S. H. G. Salem, A. Y. Hassan, M. S. Moustafa, and M. N. Hassan, "Blockchain-based biometric identity management," *Cluster Computing*, vol. 27, pp. 3741–3752, 2024. DOI: [10.1007/s10586-023-04180-x](https://doi.org/10.1007/s10586-023-04180-x).
9. Z. Su, "Face recognition research: A history of advancement," *Highlights in Science, Engineering and Technology, TPCEE 2023*, vol. 72, 2023.

10. A. Ben Said, "Multispectral imaging and its use for face recognition : sensory data enhancement," *University of Burgundy*, Thesis, 2015
11. M. A. N. U. Ghani, K. She, M. A. Rauf, S. Khan, J. A. Khan, E. A. Aldakheel, and D. S. Khafaga, "Enhancing security and privacy in distributed face recognition systems through blockchain and GAN technologies," *CMC*, vol. 79, no. 2, pp. 2609–2623, 2024. DOI: 10.32604/cmc.2024.049611.

CHAPTER 2:

1. A. S. Albaour, Y. A. Aburawi, and A. R. Shaaib, "Blockchain: Review paper," *International Journal of Academic Research in Innovative Engineering*, vol. 9, no. 5, 2023.
2. S. R. Gopireddy, "Implementing zero-knowledge proofs for enhanced security in cloud authentication systems," *European Journal of Advances in Engineering and Technology*, vol. 9, no. 11, pp. 147-150, 2022.
3. Ghafourian, M., Sumer, B., Vera-Rodriguez, R., Fierrez, J., Tolosana, R., Moralez, A., & Kindt, E. (2023). Combining Blockchain and Biometrics: A Survey on Technical Aspects and a First Legal Analysis. arXiv preprint arXiv:2303.12345.
4. M. D'Eusano and L. Petti, "Blockchain technology and Social Life Cycle Assessment: synergies and implications," *The International Journal of Life Cycle Assessment*, 2024.
5. H. Du, H. Shi, D. Zeng, X.-P. Zhang, and T. Mei, "The elements of end-to-end deep face recognition: A survey of recent advances," *Preprint arXiv:2009.13290v4*, 2020.
6. B. Xu, T. Agbele, and R. Jiang, "Biometric blockchain: A better solution for the security and trust of food logistics," *IOP Conference Series: Materials Science and Engineering*, vol. 646, p. 012009, 2019.
7. Z. Su, "Face recognition research: A history of advancement," *Highlights in Science, Engineering and Technology, TPCEE 2023*, vol. 72, 2023.
8. The Crypto Ape. "Timeline and History of Blockchain Technology." *The Crypto Ape*, [march 2nd 2022], www.thecryptoape.com/timeline-and-history-of-blockchain-technology.

9. L. Li, X. Mu, S. Li, and H. Peng, "A Review of Face Recognition Technology," *IEEE Access*, vol. 8, pp. 76117-76144, Jul. 2020, doi: 10.1109/ACCESS.2020.3011028.
10. S. Sharma, "A comprehensive study of cryptographic hash functions," *Conference Paper, Defence Materials & Stores Research & Development Establishment (DRDO)*, 2024.
11. A. A. Ali, M. Q. Ali, and M. H. Zwayyer, "A Proposed Block chain-Based Digital-Identity Management using ERC 725/735," *International Journal of Advances in Scientific Research and Engineering*, vol. 10, no. 7, 2024.
12. Richard, Y. Heryadi, Lukas, and A. Trisetyarso, "Leverage from Blockchain in Commodity Exchange: Asset-Backed Token with Ethereum Blockchain Network and Smart Contract," *Smart Trends in Computing and Communications: Proceedings of SmartCom 2020. Smart Innovation, Systems and Technologies*, vol 182, 2021.
13. R. Zhao, C. Xu, Z. Zhu, and W. Mo, "A Blockchain-Based Secure Sharing Scheme for Electrical Impedance Tomography Data," *Mathematics*, vol. 12, no. 7, 2024.
14. M. A. N. U. Ghani, K. She, M. A. Rauf, S. Khan, M. Alajmi, Y. Y. Ghadi, and H. K. Alkahtani, "Toward robust and privacy-enhanced facial recognition: A decentralized blockchain-based approach with GANs and deep learning," *Mathematical Biosciences and Engineering*, vol. 21, no. 3, pp. 4165–4186, 2024.
15. V. S. Preiya, V. D. A. Kumar, R. Vijay, V. K., and N. Kirubakaran, "Blockchain-based e-voting system with face recognition," *Fusion: Practice and Applications*, vol. 12, no. 1, pp. 53-63, 2023.
16. S. H. G. Salem, A. Y. Hassan, M. S. Moustafa, and M. N. Hassan, "Blockchain-based biometric identity management," *Cluster Computing*, vol. 27, pp. 3741–3752, 2024. DOI: [10.1007/s10586-023-04180-x](https://doi.org/10.1007/s10586-023-04180-x).

CHAPTER 3:

Document web:

1. [What is Unified Modeling Language \(UML\)?](#)

2. [Activity Diagrams - Unified Modeling Language \(UML\) - GeeksforGeeks](#)
3. [Class Diagram | Unified Modeling Language \(UML\) - GeeksforGeeks](#)
4. [Use Case Diagram - Unified Modeling Language \(UML\) - GeeksforGeeks](#)
5. [Sequence Diagrams - Unified Modeling Language \(UML\) - GeeksforGeeks](#)

CHAPTER 4:

Document web:

1. [Visual Studio Code - Code Editing. Redefined](#)
2. [Remix - Ethereum IDE & community](#)
3. [IPFS Documentation | IPFS Docs](#)
4. [Alchemy - Infrastructure Tools - Alchemy](#)
5. [Welcome to Python.org](#)
6. [Home | Solidity Programming Language](#)
7. [PyQt6·PyPI](#)
8. [PyTorch](#)
9. [web3·PyPI](#)
10. [opencv-python·PyPI](#)
11. [NumPy](#)
12. [requests·PyPI](#)
13. [IPFS: Building blocks for a better web | IPFS](#)
14. [What is Sepolia? A Beginner's Guide to Ethereum Test Networks | by GetBlock | Medium](#)

BMC, BP



1. Business Description

Overview: SecureFace is a pioneering facial recognition system that integrates blockchain technology with IPFS decentralized storage to create a secure, immutable, and transparent identity verification platform. Initially focusing on Mostaganem, the project aims to expand across Algeria.

2. Customer Segments

Who are our most important customers? Who are we directing value to?
Our potential clients are:

- Government and public sector (e.g., national ID systems)
- Banks and financial institutions (KYC/AML compliance)
- Hospitals and insurance providers
- Remote onboarding services (e.g., fintech, e-health)

3. Customer Relationships

How we attract customers' attention to your products or services? How to encourage a customer to buy our product or service?

How does the customer benefit from our product or service?

- Personal Assistant: Dedicated support for users.
- Regular Communication: Updates on new features and improvements.
- Feedback Collection: Surveys and direct communication to address user concerns..

4. Value Proposition

What value do we offer to the customer?

How do we help the customer resolve their issues?

What is the nature of this solution to the problem, are they qualitative or quantitative values?

- Tamper-proof identity verification system
- Enhanced privacy and security through decentralization
- Immutable audit trails for access logs
- Reduction in identity fraud and increased trust
- Suitable for remote verification (e.g., onboarding, border control).

5. Key Activities

What are the most important stages of production or service? (We mention the steps from the acquisition of raw materials to the final product)

- Facial recognition model development
- Blockchain integration (smart contracts, data validation)
- IPFS connection for decentralized storage
- GUI interface development

6. Key Partners

Who are the main partners who can help us in production or service or in marketing or distribution?

Who are the main suppliers? (Those who supply us: raw materials + machines for production + software for service +...)

- Blockchain infrastructure providers (Ethereum, Hyperledger, IPFS, etc.)
- Suppliers of cameras and edge devices for facial recognition

7. Channels

How does the public know about us, our product or our service?

Which distribution channels do customers prefer to communicate with? What are the most cost-effective channels?

- Collaborative platforms with partners
- Awareness campaigns (e.g., digital marketing)
- Free trial periods for evaluation
- Pre-sales services (demos, webinars, training)

8. Revenue Streams

- Licensing fees for enterprise use
- Pay-per-verification API access
- Monthly SaaS subscriptions
- Custom integration services

9. Cost Structure

This part outlines the various costs that our Platform will incur

- Hosting blockchain nodes and databases
- Data storage (encrypted biometric data)
- Developer salaries (AI, blockchain, full-stack)
- Compliance and legal costs
- Marketing and outreach.

10. Key Resources

- Facial recognition datasets and models
- Blockchain platform (Ethereum, Hyperledger, etc.)
- Skilled team in blockchain and AI technologies

10.1 Material resources :

Element	Price	Qty	Total
Developer Laptops	600,000.00 DA	2	1200,000.00 DA
Office Setup	150,000.00 DA	1	150,000.00 DA
modem	10000.00 DA	1	10000.00 DA
Total CAPEX			1360,000.00 DA

10.1 charges:

Element	Cost
Rent (Co-working Space)	30,000.00 DA/month
Utilities	5,000.00 DA/month
Internet	4,000.00 DA/month
Public Blockchain Fees	
› Ethereum L2 (Polygon)	\$50/month ≈ 6,750 DA
› Filecoin (IPFS Pinning)	\$30/month ≈ 4,050 DA

Cloud Backend (AWS Free Tier)	\$0 (first year)
Total Monthly OPEX	145,800.00 DA

Startup Financial Summary
Phase 1: Bootstrap Mode (Year 1)
material resources: 1,360,000.00 DA
charges: 1,748,600 DA
Total Year 1 Cost: 3,108,600 DA

Element	Purchase price	Cycle	Qty	amortization
PC	600000.00DA	03	2	400 000.00
Office Setup	150000.00DA	04	1	37,500.00DA
modem	10000.00DA	02	2	5 000.00DA

الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي

جامعة عبد الحميد بن باديس - مستغانم
كلية العلوم الاجتماعية والإنسانية

تصريح شرفي خاص بالالتزام بقواعد النزاهة العلمية
لاتجار البحث

أنا الممضي أدناه،

المثبت: (أ) حسن بن خالطية الزهرية 192037037270 رقم تسجيل
الجامعة: 192037037270

الحامل لبطاقة التعريف الوطنية رقم 1522000915220009 والصادرة بتاريخ:

..... 16.01.2019

عن... صواب... مستغنا... من...

المسما... بكلية العلوم الدقيقة والاعلام... في...

شعبة... إلى... التخصص

الدراسة... في... في...

والمكلف بإتجار متكرة ماستر بعنوان:

Integrating Blockchain in Facial Recognition
Systems

أصرح بشرفي أنني ألتزم بمراعاة المعايير العلمية والمنهجية ومعايير الأخلاقيات العلمية
والنزاهة الأكاديمية المطلوبة في إتمام البحث ، وأتحمل المسؤولية الشخصية عن كل
تصريح شخصي في البحث المذكور أعلاه .

التاريخ 23 - 06 - 2025

