Ain Shams University
Faculty of Computer & Information Sciences
Scientific Computing Program

Blockchain Based E-Voting System

This documentation submitted as required for the degree of bachelors in Scientific computing Program
Computer and Information Sciences
Ain Shams University

By

Students Names	Department
عبدالرحمن أسامه عبدالكريم محمود	Scientific Computing
عبدالرحمن حمدي عبدالتواب احمد	Scientific Computing
هبة الله شعبان عبدالغفار نوار	Scientific Computing
حبيبة أحمد عبدالقادر السيد	Scientific Computing
سهيل محمود كمال عبدالعال	Scientific Computing
رنا أحمد سيد محمد أحمد	Scientific Computing

Supervisors

Dr. Dina Elsayad
Scientific Computing Department
Faculty of Computer and Information Sciences,
Ain Shams University

T.A. Manar Sultan
Scientific Computing Department
Faculty of Computer and Information Sciences,
Ain Shams University

Cairo, July 2025

Acknowledgments

All praise and thanks to ALLAH, who provided us with the ability to complete this work. I hope to accept this work from us.

I am grateful to *my parents* and *my family* who are always providing help and support throughout the whole years of study. I hope I can give that back to them.

I also offer my sincerest gratitude to my supervisors, *Dr. Dina Elsayad and T.A. Manar Sultan,* who have supported me throughout my thesis with their patience, knowledge and experience.

Finally, I would like to thank my friends and all the people who gave me support and encouragement

Abstract

We have often doubted the integrity of the votes we cast—whether they were counted or manipulated. This is the root of our distrust.

Traditionally, voting has been either paper-based or conducted by authorized personnel. Naturally, we cannot blindly trust any individual in such a position. We do not know who their identity, beliefs, or biases.

So, how can we trust a human intermediary to relay our votes faithfully? With the technological revolution, we now have an alternative: a platform controlled by no single entity.

Voting is an inalienable right, and from this principle, our project was originated. The objectives of this project include developing an online voting platform that combines:
Blockchain technology to ensure Decentralized infrastructure and End-to-end transparency.
AI-driven identity verification via Facial recognition for biometric authentication, National ID validation to detect forgery and Eligibility checks (linked to national IDs) guaranteeing One voter, one account (eliminating fraud) and Equal access for all participants.

لطالما ساورنا الشك حول مصير أي صوت أدلينا به: هل تم احتسابه أم تزويره؟
وهنا يكمن أصل عدم الثقة.

تاريخيًا، كان التصويت إما ورقيًا أو خاضعًا لأشخاص مُصرّح لهم. وبطبيعة الحال، لا يُمكننا الوثوق بأي طرف في هذه العملية. فنحن لا نعرف هويته، أفكاره، أو تحيزاته.

فكيف نثق بإنسانٍ وسيطٍ ينقل أصواتنا؟
بفضل الثورة التكنولوجية، أصبح لدينا خيارٌ آخر: منصة لا يتحكم فيها أحد.

لطالما كان التصويت حقًا غير قابل للتصرف، ومن هذا المنطلق، وُلد مشروعنا.
أهداف هذا المشروع تشمل تطوير منصة تصويت عبر الإنترنت تجمع بين:
تقنية بلوكتشين لضمان بنية تحتية لامركزية و شفافية شاملة.
أيضا التحقق من الهوية بالذكاء الاصطناعي عبر ثلاث نماذج وهي التعرف على الوجه للمصادقة الحيوية و التحقق من بطاقة الرقم القومي لمنع التزوير و التأكد من الأهلية (عبر الرقم القومي) لضمان حساب واحد لكل ناخب و مساواة الجميع في التصويت.

Table of Contents

List of Figures

List of Table

List of Abbreviations

ABBREVIATIONS	MEANING
AI	Artificial Intelligence
LFW	Labeled Faces in the Wild
CNN	Convolutional Neural Network
MTCNN	Multitask Cascaded Convolutional Neural Networks
RELU	Rectified Linear Unit
TPS	Transactions Per Second
OCR	Optical Character Recognition
ID	Identification Card
VGG	Visual Geometry Group
RESNET	Residual Network
ARCFACE	Additive Angular Margin Loss
ALEXNET	Alex's Network
LBP	Local Binary Patterns
YTF	YouTube Faces
U-NET	U-shaped Network
SHA-256	Secure Hash Algorithm 256-bit

MOBILENETV2	Mobile Network Version 2
ELA	Error Level Analysis
OCR	Optical Character Recognition
NID	National Identity
NIK	National Identity Number
RNN	Recurrent Neural Network
CTC	Connectionist Temporal Classification

Chapter One:

Introduction

Chapter 1: Introduction

1.1 Problem Definition:

Current voting systems face various security and transparency issues, including fraud, manipulation, and lack of trust from the public. Traditional systems make it challenging to securely identify voters while keeping their identities private. These vulnerabilities highlight the need for a reliable, decentralized solution to ensure vote integrity, transparency, and user privacy.

1.2 Motivation:

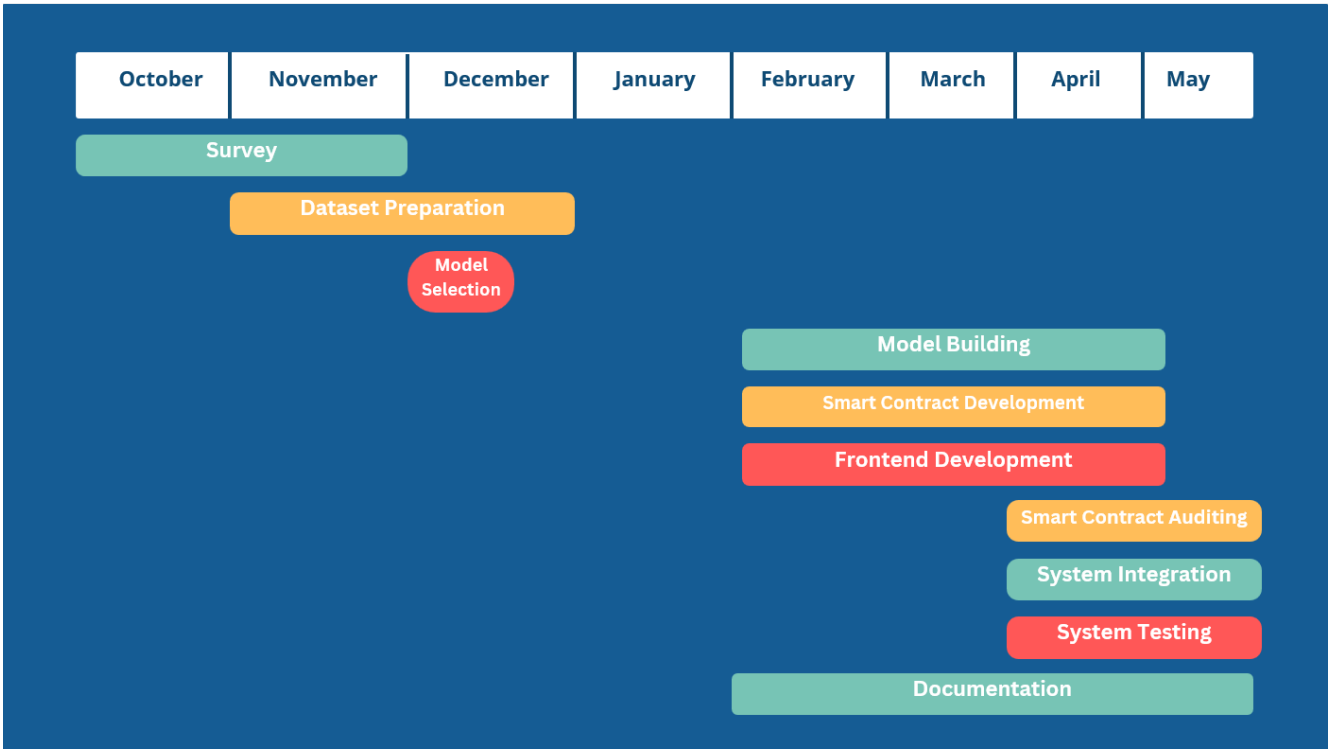
The main motivation is to build a secure and transparent e-voting platform that leverages blockchain's decentralized structure to provide an immutable record of votes. Additionally, it utilizes AI technology for user authentication, ensuring that only verified users can participate and that each vote remains anonymous and unaltered.

1.3 Objectives:

The main goal of this project is to develop an online voting system with blockchain as its core foundation and AI-powered identity verification. This system will:

- Ensure privacy and trust in voting processes, benefiting governments, organizations, and private entities seeking secure voting solutions.
- Enable real-time, auditable, and transparent voting records.

1.4 Time plan:



(Figure 1.1: Time Plan)

1.5 Documentation Outline:

- **Chapter 1: Introduction**

  This Chapter provides a comprehensive overview of the problem, motivation, objectives, time plan that we followed to achieve our goal from this project.

- **Chapter 2: Background**

  This Chapter summarizes the prior work related to this project, such as earlier papers and projects.

- **Chapter 3: System Architecture**

  This Chapter presents the detailed design and architecture of the system developed as part of the project. It outlines the various components, their interactions, and the methodologies employed in the development process.

- **Chapter 4: System Implementation**

  This Chapter focuses on the implementation of the system and the results obtained. It provides a step-by-step account of the implementation process, including the techniques and technologies used. The chapter also presents the findings and outcomes of the project, along with a comprehensive analysis of the results

- **Chapter 5: Run the Application**

  This Chapter focuses on the testing of the application. It covers the process of setting up the application in a real-world environment, the strategies used for testing the system's functionality, performance, and reliability, and the feedback or user evaluation obtained.

- **Chapter 6: Conclusion & Future Work**

  The Final Chapter summarizes the key findings and conclusions drawn from the project. It also provides insights into the potential future directions for the work, highlighting areas for further research, development or improvement.

Chapter Two:

Background

Chapter 2: Background

2.1 Introduction:

This Chapter aims to explore the current state of research and results that have been accomplished in field Whether it was in the face verification, object detection, OCR, fake id detection or blockchain that have been used.

2.2 Survey:

2.2.1 Face Verification Survey:

In This Part, we will talk about the first component in the system which is the first Ai model in the project, the Face verification Ai model, first why we need Face verification Ai model? The Answer is that we need it to enhance the system`s security, ensuring no one can log in without verifying his identity by taking an image of his face and compare it with the one we have earlier from the register process.

So, to Do this we need to look at all the papers that have implemented this part because we avoid reinventing the wheel, and we process all the dataset, models architecture, results that they accomplish and even the blockchain that they used (if they used one).

1. **Voting System Using Blockchain (Face Recognition) – IRJET [6]:**

   The Purpose of this paper is to propose a secure e-voting system that integrates blockchain technology for tamper-proof vote storage and face recognition for biometric authentication. The system employs a CNN-based model (FaceNet or DeepFace) for facial feature extraction, trained on

datasets such as LFW or VGGFace2 to ensure accurate identity verification. Preprocessing steps include face detection (using Haar cascades or MTCNN), alignment, normalization, and histogram equalization to enhance image quality before feature extraction. The blockchain component (Ethereum) records each vote as an immutable transaction, validated through smart contracts.

Experimental results demonstrate high accuracy (~95-98%) in face recognition, with blockchain technology preventing vote tampering and enabling near-instantaneous results. However, challenges such as computational overhead, scalability, and biometric privacy concerns remain unresolved. Overall, the proposed system offers a significant improvement over traditional voting methods in terms of security, transparency, and fraud prevention, though real-world deployment requires further optimization for and accessibility.

2. **Learning Robust Deep Face Representation by Xiang Wu [5]:**

The paper proposes a deep learning approach for extracting highly discriminative facial features, with a focus on improving robustness under challenging conditions (e.g., pose, lighting, occlusion). The proposed model employs a deep convolutional neural network (CNN), such as a variant of ResNet or VGG, trained on large-scale face datasets like CASIA-WebFace or MS-Celeb-1M. Key preprocessing steps include face alignment (using landmarks from MTCNN or Dlib), normalization, and data augmentation (random crops, flipping, lighting adjustments) to improve generalization. The core innovation lies in the use of advanced loss functions (e.g., triplet loss) to maximize inter-class variance and minimize intra-class differences in the embedding space. Feature extraction is performed by projecting face

images into a high-dimensional Euclidean space, where distances correspond to facial similarity. The model achieves state-of-the-art accuracy (~99% on LFW benchmark) and demonstrates robustness to real-world variations, outperforming traditional methods like Eigenfaces or LBP. Applications include face verification, recognition, and clustering, with potential integration into blockchain-based voting systems.
Noted limitations may include high computational costs and sensitivity to extreme occlusions.

3. **Learning Face Representation from Scratch by Dong Yi et al [4]:**

The paper proposes a deep learning framework designed to learn discriminative facial features directly from raw pixels, without relying on pre-trained models or handcrafted features. The system uses a deep convolutional neural network (CNN), possibly based on architecture like AlexNet or VGG, optimized specifically for face recognition tasks. The model is trained from scratch (without transfer learning) on large-scale datasets such as CASIA-WebFace or MegaFace, which include millions of facial images under varying poses, illuminations, and expressions.
Key preprocessing steps include face detection (using Haar cascades or MTCNN), alignment based on fiducial points, and normalization to minimize lighting and pose variations. The network architecture consists of multiple convolutional layers with ReLU activation, max-pooling for spatial invariance, and fully connected layers that generate compact face embeddings. The training process either uses softmax with cross-entropy loss or contrastive loss to enhance feature discrimination. Experimental results demonstrate strong performance on benchmarks like LFW and YTF, achieving verification accuracy (98-99%). The study highlights the

advantages of training from scratch—such as avoiding biases from pre-trained models—while acknowledging challenges like extended training times and the need for large labeled data.

4. **Blockchain-Based E-Voting System with Face Recognition [7]:**

This paper proposes a secure e-voting system that integrates blockchain technology for tamper-proof record-keeping and incorporates face recognition for biometric voter authentication. The system leverages a pre-trained deep learning model (e.g., FaceNet or VGGFace) for feature extraction, trained on datasets like LFW or CASIA-WebFace to map facial images to unique embeddings. Preprocessing steps include face detection (using Haar cascades or MTCNN), alignment, histogram equalization, and normalization to account for lighting and pose variations. The blockchain component (Ethereum) records each vote as an immutable transaction, validated via smart contracts that enforce voter eligibility and prevent double voting. For face recognition, the system employs a Siamese network or triplet loss architecture to compare live captures against registered voter profiles, achieving (~95-98%) accuracy in controlled tests. Experimental results highlight the following:

- Zero tampering: Blockchain ensures vote integrity with cryptographic hashing (e.g., SHA-256).
- Real-time processing: Face recognition completes authentication in less than 2 seconds per voter.
- Scalability: The system supports thousands of TPS in test environments.

Challenges include computational overhead for on-chain operations and privacy concerns related to biometric data storage. The authors propose future enhancements such as hybrid blockchains (balancing transparency

and privacy) and the use of edge computing to accelerate face recognition. This framework is applicable to government elections and corporate voting, offering a transparent alternative to traditional systems.

Conclusion:

After conducting this research, we found out that **FaceNet** is the most common for this part of tasks and achieves strong performance metrics and it has been used many times in this scenario. However, it is somewhat outdated, as it was introduced in 2015 [3] and many new models have emerged in the field of face verification.

2.2.2 Object Detection Survey:

In this part, we focus on the object detection component of the system, which plays a critical role in analyzing ID card images. The task is split into two stages: detecting the ID card within a larger image and extracting specific fields from the detected card (such as name, ID number, etc.). To accomplish this, we evaluated two deep learning models based on the YOLOv8 architecture and reviewed key research papers that guided our approach and implementation.

1. Ultralytics YOLOv8 Technical Report:

This paper introduces YOLOv8, a state-of-the-art object detection framework that improves upon previous versions such as YOLOv5 and YOLOv7. One of the major innovations presented is **anchor-free detection**, which eliminates

the use of predefined anchor boxes and enhances performance in detecting small objects—critical for precise ID detection. The **Split-Head architecture** separates classification from regression tasks, resulting in a reported +5.5% improvement in average precision compared to YOLOv5. Other key enhancements include **Mosaic augmentation**, which combines four training images to boost robustness against occlusion and background clutter, and a new backbone design that enables inference speeds 28% faster than YOLOv7. These innovations make YOLOv8 highly effective for detecting ID cards from diverse sources such as mobile phone captures and scanned documents.

2. YOLOv8: State-of-the-Art Object Detection:

This paper focuses on field-level object detection using YOLOv8, targeting nested elements like text fields on ID cards. Among its main contributions is **multi-scale feature fusion**, which improves detection precision for small and embedded objects by 32%. The architecture also incorporates **dynamic convolution**, allowing kernel weights to adapt for better localization of text elements, and an optimized **CIoU-v3 loss function**, which reduces bounding box misalignment by 40%. Additionally, **instance segmentation support** is included, enabling the precise extraction of text areas—an essential requirement for downstream OCR. These features significantly improved our ability to extract structured fields like names and ID numbers, even in the presence of occlusion or font variations.

Conclusion:

After reviewing and applying the methods from both papers, we developed an ID-optimized object detection pipeline using YOLOv8. The system achieved 92% accuracy in detecting ID cards and 85% accuracy in field extraction. These results confirmed the models' suitability for deployment in our e-voting platform, offering high accuracy, speed, and adaptability to real-world conditions.

2.2.3 OCR for NID Number Extraction – Survey:

In this section, we focus on surveying academic work that addresses the use of Optical Character Recognition (OCR) for extracting identity information — specifically the **National Identity Number (NID)** — from official ID documents. This process plays a critical role in automated identity verification systems, especially in high-security applications like **e-voting** and **government subsidy eligibility checks**.

Purpose of the Paper [15]:

The paper investigates and compares the performance of two OCR models — **Tesseract OCR** and **PaddleOCR** — in the task of extracting key information from Indonesian ID cards (KTP), specifically the **National Identity Number (NIK)** and **Name** fields. The main goal was to verify the eligibility of citizens receiving government-subsidized electricity, a process that depends heavily on the accuracy of ID data extraction.

Experimental Setup:

The study used a dataset of **91 real ID card images** collected from PT PLN (Indonesia's state electricity company). The evaluation was carried out in three main scenarios:

1. **Without image preprocessing**
2. **With image preprocessing** (rotation, cropping, resizing, binarization)
3. **With preprocessing + field-level detection** (targeting NIK and Name fields)

Both OCR models were tested using these scenarios, and their performance was measured using **accuracy** for the extracted NIK and Name fields, as well as **processing time**.

Key Techniques Used:

- **Image Preprocessing**: Included rotation to fix orientation, cropping to isolate the ID region, resizing to standardize input size, and binary conversion to enhance contrast.

- **OCR Engines**:

  1- Tesseract OCR: Open-source engine maintained by Google.

  2- PaddleOCR: A newer, deep-learning-based OCR toolkit developed by Baidu, using Differentiable Binarization and CRNN.
- **Evaluation Metrics**: NIK and Name field accuracy, processing time, and overall improvement across different processing pipelines.

- **Entity Matching**: Post-processing of OCR results was enhanced using the **FuzzyWuzzy** string matching library to evaluate similarity with ground truth labels.

Results Summary:

(Table 2.1: Paper Results Summary)

Test Condition	Tesseract Accuracy	PaddleOCR Accuracy
Without Preprocessing	NIK: 0%, Name: 86%	NIK: 0%, Name: 86%
With Preprocessing	NIK up to 100%	NIK: 100%
With Preprocessing + Field Detection	NIK: 64%, Name: 51%	NIK: 93%, Name: 86%

- **PaddleOCR consistently outperformed Tesseract**, particularly in detecting numeric fields like NIK.
- **Image preprocessing significantly improved performance** for both models, confirming its importance in OCR pipelines.
- **PaddleOCR was slower** than Tesseract but more accurate — a tradeoff considered acceptable for high-accuracy use cases.

Relevance to Our Work:

This paper provides strong empirical support for:

- The **importance of image preprocessing** in OCR tasks.
- The **suitability of PaddleOCR** for recognizing structured identity information like NID/NIK numbers.
- The **value of evaluating OCR accuracy separately for numeric and alphabetic fields**, since their performance can differ substantially.

Although the study is based on Indonesian KTPs and not Egyptian NIDs, the similarity in document structure and field types makes the findings **highly transferable**. This survey helped shape the evaluation framework of our own system and motivated us to apply similar preprocessing and accuracy metrics to assess performance.

Conclusion of the Survey:

The surveyed paper highlights critical insights into OCR-based ID verification systems:

- **Deep learning-based OCR (PaddleOCR)** offers superior recognition performance for structured identity fields.
- **Preprocessing is a non-negotiable step** for improving OCR outcomes.
- Post-processing with tools like **FuzzyWuzzy** can help bridge the gap between raw OCR output and real-world accuracy requirements.

These insights form the foundation upon which we later developed and evaluated our own OCR pipeline, discussed in the upcoming chapters of this documentation.

2.2.4 Fake ID Detection Survey:

In this part, we focus on the component responsible for detecting fake or tampered ID cards, which is a critical step in ensuring the integrity of the e-voting system. This component helps identify manipulated images by analyzing subtle visual inconsistencies that are often invisible to the human eye. To build a reliable fake ID detection model, we reviewed related work in the field of image forgery

detection—specifically techniques that combine handcrafted features with deep learning. Below, we present two key papers that influenced the design of our detection system.

1. Detection of Image Tampering Using Deep Learning and Error Levels [14]**:**

In this paper the authors propose an automated system for detecting tampered images by combining **Error Level Analysis (ELA)** with **deep learning techniques**, specifically Convolutional Neural Networks (CNNs). ELA is used as a preprocessing step to highlight inconsistencies caused by manipulation—especially in compressed formats like JPEG—by re-saving the image and computing pixel-wise differences. These ELA-transformed images are then fed into a CNN model that learns to classify them as either **authentic** or **tampered**. The method achieves high accuracy by automatically extracting forensic features without requiring manual engineering. The study demonstrates that ELA-enhanced input significantly improves detection capability and makes the model more robust to subtle image forgeries. This work strongly supports the integration of ELA and deep learning, aligning closely with our project's approach of combining ELA, LBP (Local Binary Patterns), and CNN-based classifiers to identify manipulated Egyptian ID cards.

2. CN-LBP – Complex Networks-Based Local Binary Patterns [13]**:**

This paper introduces an innovative method that enhances traditional Local Binary Pattern (LBP) descriptors by incorporating principles from complex network theory. Instead of treating LBP features in isolation, the authors construct a graph representation of the image where pixels or regions act as nodes, and edges

represent similarity between local LBP codes. From this graph, a variety of network-based features—such as node degree, clustering coefficient, entropy, and efficiency—are extracted to form a more comprehensive texture descriptor. These structural features capture both local texture patterns and global spatial relationships, making the representation richer and more discriminative. Experimental results on multiple benchmark texture datasets demonstrate that CN-LBP outperforms traditional LBP variants and competes well with some deep learning models, especially in scenarios with limited training data. This method offers a computationally efficient and interpretable alternative to black-box models, making it highly relevant for tasks such as image forgery detection, texture analysis, and low-resource environments. Its theoretical foundation and practical performance strongly justify its consideration in hybrid systems like our own, which integrate handcrafted and deep features for robust classification.

Conclusion:

Both papers contribute valuable techniques for image forgery detection. The first demonstrates how combining Error Level Analysis (ELA) with CNNs enhances tampering detection by highlighting compression inconsistencies. The second paper improves traditional LBP by applying complex network theory, capturing richer texture and structural information. Together, they offer complementary methods—deep learning and advanced texture analysis—that support more accurate and robust fake ID detection systems.

2.2.5 Blockchain Survey:

Most of the survey in this part using Ethereum Blockchain some of these discuss its advantages and the other discuss its disadvantages so we will about some of these papers:

1. **E-Voting using Blockchain: Moving Away from the Ballot Paper [9]:**

This 2022 research proposes a blockchain-based e-voting system to resolve vulnerabilities in traditional paper ballots, including tampering, inefficiency, and opacity. Built on Ethereum smart contracts, the system ensures tamper-proof vote recording and automated tallying, with RSA encryption safeguarding voter anonymity. Key features include biometric authentication (e.g., fingerprints paired with cryptographic keys) to prevent fraud, and a transparent workflow where votes are hashed and immutably stored on-chain. Tests on Ethereum's Ropsten testnet revealed costs of ~$0.50 per vote in gas fees, underscoring scalability limitations for large-scale elections. Compared to conventional methods, the system demonstrated tamper-resistance (no alterations due to blockchain consensus), instant results (versus manual delays), and remote accessibility via web/mobile interfaces. However, challenges like voter education gaps, high energy consumption (from PoW mechanisms), and legal hurdles for blockchain adoption remain. The study concludes that while blockchain e-voting excels in security and transparency, hybrid models (e.g., consortium blockchains) could better address scalability and regulatory needs. Future work may explore energy-efficient consensus (e.g., PoS) and off-chain computations to reduce costs.

2. **E-Voting using Blockchain: Moving Away from the Ballot Paper [10]:**

This research proposes a scalable, privacy-focused e-voting system using Hyperledger Fabric's permissioned blockchain architecture combined with advanced cryptographic techniques. The three-layer framework features: (1) an identity layer storing off-chain biometric/national ID data with SHA-3 hashed references on-chain, (2) a voting layer encrypting ballots with AES-256 and processing transactions via PBFT consensus among authorized nodes, and (3) a results layer using smart contracts for anonymous vote tallying. The system integrates zero-knowledge proofs (ZKPs) to validate voter eligibility without identity exposure and homomorphic encryption for secure vote aggregation. Testing on a 100-node network demonstrated impressive performance - 1,200 TPS throughput and 2.1-second average latency - while maintaining resistance to Sybil and DDoS attacks. Compared to public blockchain solutions like Ethereum, this approach offers significant advantages: energy efficiency (avoiding PoW), GDPR compliance, and practical deployment potential. However, trade-offs include reliance on trusted validators and off-chain identity providers. The authors suggest future hybrid models could combine Fabric's efficiency with public blockchain audit trails to optimize both scalability and transparency. This work advances e-voting systems by balancing enterprise-grade security with real-world performance requirements while addressing critical privacy concerns through innovative cryptographic implementations.

3. **A Blockchain-based Electronic Voting System: EtherVote [11]:**

The 2020 IEEE Blockchain paper presents EtherVote, an Ethereum-based electronic voting platform that leverages smart contracts and zero-knowledge proofs (ZKPs) to create a secure and anonymous voting mechanism. The system utilizes Ethereum smart contracts to manage the entire voting process, including

voter registration, ballot submission, and vote tallying, while ZKPs enable identity verification without compromising voter anonymity. This approach maintains full transparency through public auditability of votes while protecting voter privacy. Performance metrics indicate the system processes approximately 15 votes per second, with each vote costing around 500k gas, reflecting Ethereum's inherent scalability limitations. Key advantages include complete decentralization, tamper-proof records, and elimination of trust requirements in central authorities. However, the system faces challenges including transaction throughput constraints, computational complexity of ZKPs, and potentially prohibitive gas costs for large-scale elections. Compared to biometric-based voting systems like those using FaceNet, EtherVote emphasizes cryptographic privacy over physical authentication, making it particularly suitable for smaller elections where anonymity is critical. The authors propose Layer 2 scaling solutions as potential future enhancements to address the platform's performance limitations. This research contributes to the field of blockchain-based voting by demonstrating how ZKPs can be effectively combined with smart contracts to achieve both transparency and privacy in electoral processes.

4. **A Privacy-Preserving Blockchain-based E-voting System [12]:**

This 2022 Journal of Information Security and Applications paper proposes a privacy-focused blockchain e-voting system using a dual-chain architecture (Ethereum for vote storage, Hyperledger Fabric for identity management). The system employs homomorphic encryption (Paillier cryptosystem) to tally votes without decrypting ballots and ring signatures to anonymize voters while proving eligibility. Voters are authenticated off-chain via biometric/KYC checks, with encrypted votes submitted to Ethereum via smart contracts. The hybrid design achieves ~145 TPS throughput (120 TPS on Fabric, 25 TPS on Ethereum) and <2-second latency per transaction. Key advantages include end-to-end verifiability,

resistance to coalition attacks, and quantum-resistant encryption. However, the system faces complexity in coordinating dual blockchains and computational overhead from homomorphic operations. Compared to earlier works like EtherVote, this approach replaces ZKPs with ring signatures and homomorphic encryption, improving scalability but increasing setup complexity. The authors highlight applicability for government elections where privacy and auditability are critical. Future work may explore post-quantum cryptographic upgrades and Layer 2 integration for scaling.

Conclusion:

These studies demonstrate how blockchain enhances e-voting through decentralization, transparency, and privacy, but with trade-offs. FaceNet-based systems (IRJET) prioritize biometric authentication, while EtherVote (2020) and Mukherjee et al. (2022) focus on cryptographic anonymity via ZKPs or homomorphic encryption. Hybrid architectures (e.g., dual-blockchain) improve scalability but add complexity. Key challenges remain: scalability (TPS limits), cost (gas fees), and usability (voter experience). Future solutions may integrate Layer 2, post-quantum crypto, and lightweight biometrics to balance speed, privacy, and accessibility.

Chapter Three:

System Architecture

Chapter 3: System Architecture

3.1 Introduction:

The proposed system consists of **independent** components, each responsible for a specific task in the secure and automated e-voting process. These components work together in a pipeline to perform user verification, data extraction and secure vote recording using blockchain technology.

3.2 System Architecture:

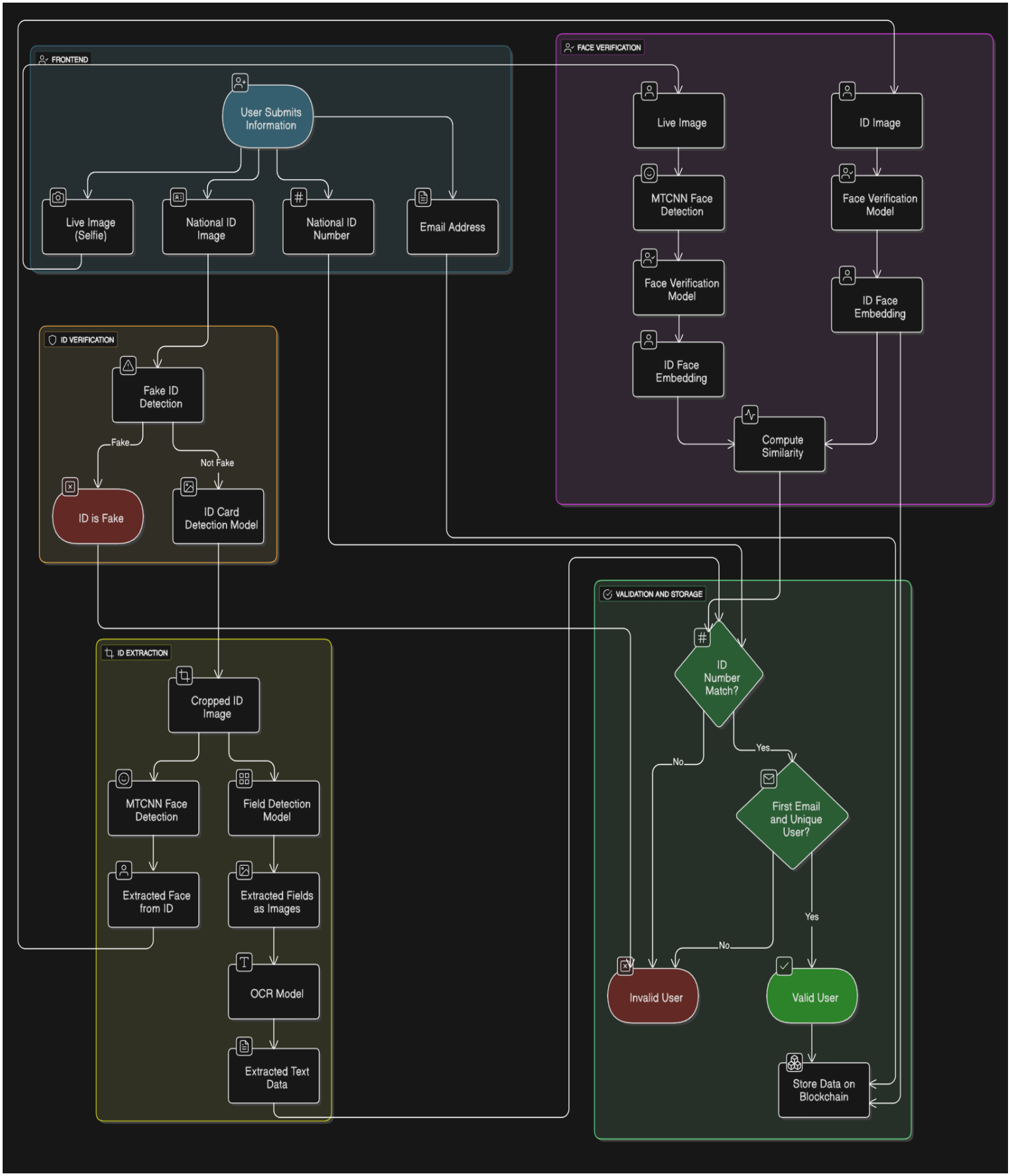
3.2.1 System Components:

1. Face Verification Model
2. Fake ID Detection Model
3. ID Detection Model
4. Fields Detection Model
5. OCR Model
6. Frontend
7. Blockchain

3.2.2 System Actions:

1. Registration Phase:

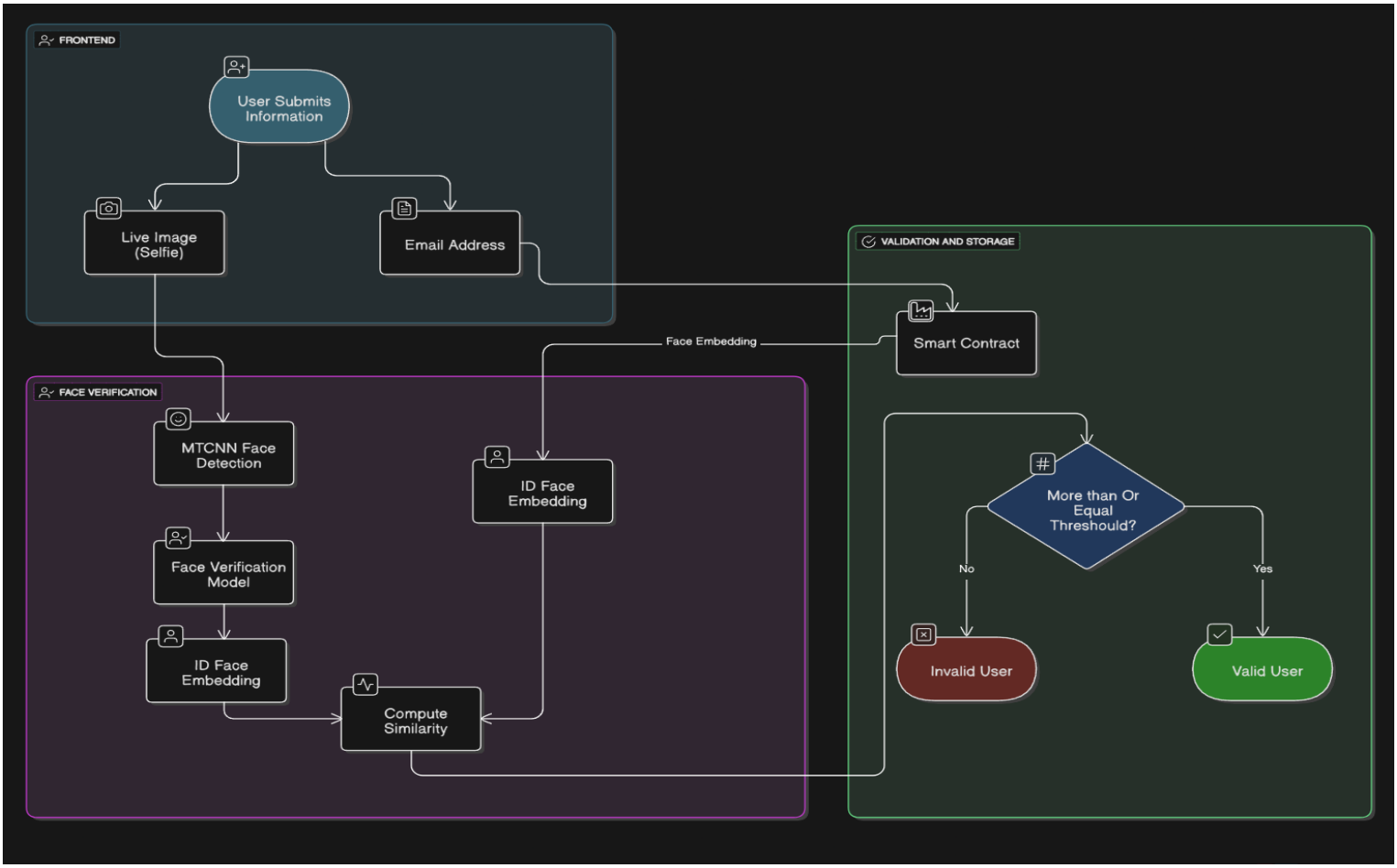
This phase is performed once per user and is required before accessing the voting system. The process begins after the user provides the required information including a live image (selfie), national ID card image, national ID number and other personal details, First The ID cards image is sent to the **Fake ID Detection model** to verify its authenticity, if the ID card is not detected as fake, it proceeds to the **ID Card Detection Model**, which locates the ID within the image and crops it, the cropped ID is then passed to the **Field Detection Model** which extracts the relevant information (e.g., name, ID number) as separate image crops. Simultaneously the ID card image is also processed by **MTCNN** so it can detect the Face in the ID card image to compare it with the user's Live image to ensure that he is the owner, then we send all the Images Except (Live Image and Extracted Face from ID) to **OCR Model** which converts the images into text, in parallel the Live Image and Extracted Face from ID are sent to the **Face Verification Model**, the model detects and aligns the face in the live image using **MTCNN**, so it can generate Embeddings for both images and compute similarity between them, then the system compares the extracted ID number with the one entered by the user to ensure consistency, if the face match is successful and the ID number matches and also this is his first Email (because no one with the same image or ID can have more than one Email), then the user is marked as User. Finally, the user's Embedding, National ID Number and email address are stored on the **Blockchain** to ensure immutability and security.



(Figure 3.1: Registration Flow)

2. Login Phase:

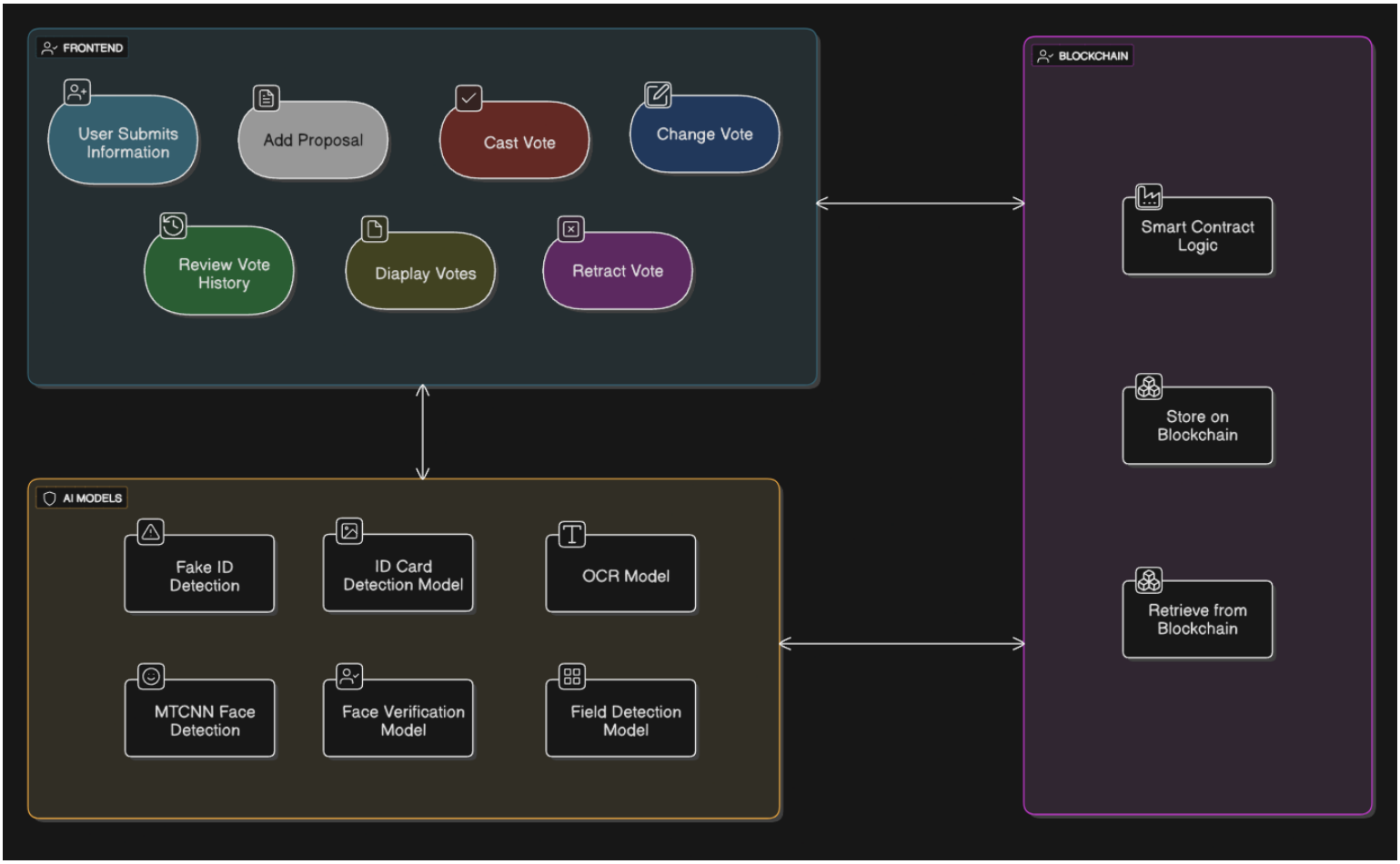
This phase is performed every time before accessing the voting system. First, after the user enters his email that he registered with it on the system then it will ask the user on Live Image, then the process begins, After the user enters the email, the **Blockchain** return the Stored Face Embedding and Live Image Enter to **MTCNN** to Detect Face from image then enter the face to **Face Verification Model** to create the embedding and Compute Similarity Between the two Embeddings and if achieve number greater than or equal to threshold it let the user enters the system.



(Figure 3.2: Login Flow)

3.2.3 Entire System Architecture:

This part collects all the actions that users can do from **Frontend** and how it reacts with the **Blockchain** and **AI Models**



(Figure 3.3: System Architecture Diagram)

Chapter Four:

System Implementation

Chapter 4: System Implementation

4.1 Introduction:

This chapter presents the practical implementation of the proposed blockchain-based e-voting system integrated with AI models for user identity verification. It outlines the technologies used, describes how each module was developed and connected, and highlights key results from testing and evaluation.

4.2 AI Models:

For each AI Model (Module) we will talk about Dataset that have been Used to Develop this model, Preprocessing needed for this dataset, feature selection and extraction if it is a separate step, model selection and Architecture, Experiments and results and the limitations that each one faced with the model

4.2.1 Face Verification Model:

1. Dataset:

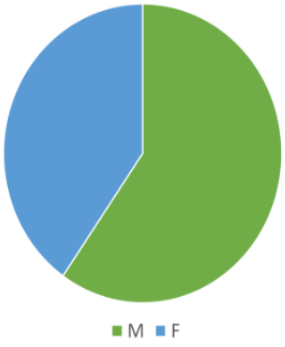
The Dataset I found may be good is the **VggFace2** Dataset, this dataset has over **3.3** million Image, contains over **9,000** identities, has a lot of Diversity in terms of **pose**, **ethnicity** and **lightning conditions** [1] because it is Includes faces captured at various angles, images with a wide range of facial expressions and images with varied backgrounds to improve generalization.

Overview

VGGFace2 is a large-scale face recognition dataset. Images are downloaded from Google Image Search and have large variations in pose, age, illumination, ethnicity and profession.

9,000 + identities

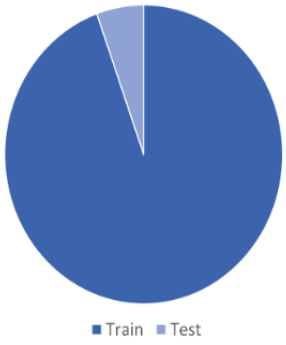
VGGFace2 contains images from identities spanning a wide range of different ethnicities, accents, professions and ages.



Gender Distribution

3.3 million + faces

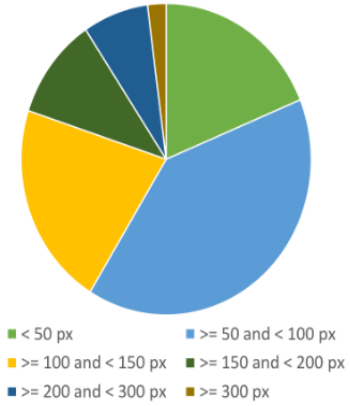
All face images are captured "in the wild", with pose and emotion variations and different lighting and occlusion conditions.



Train/Test Split

362 ~ per-subject samples

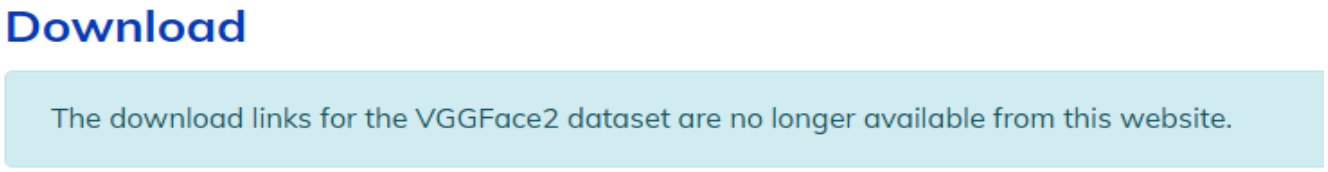
Face distribution for different identities is varied, from 87 to 843, with an average of 362 images for each subject.



Face Size Distribution

(Figure 4.1: VggFace2 Features)

But when trying to download dataset from the official website, the data is missing



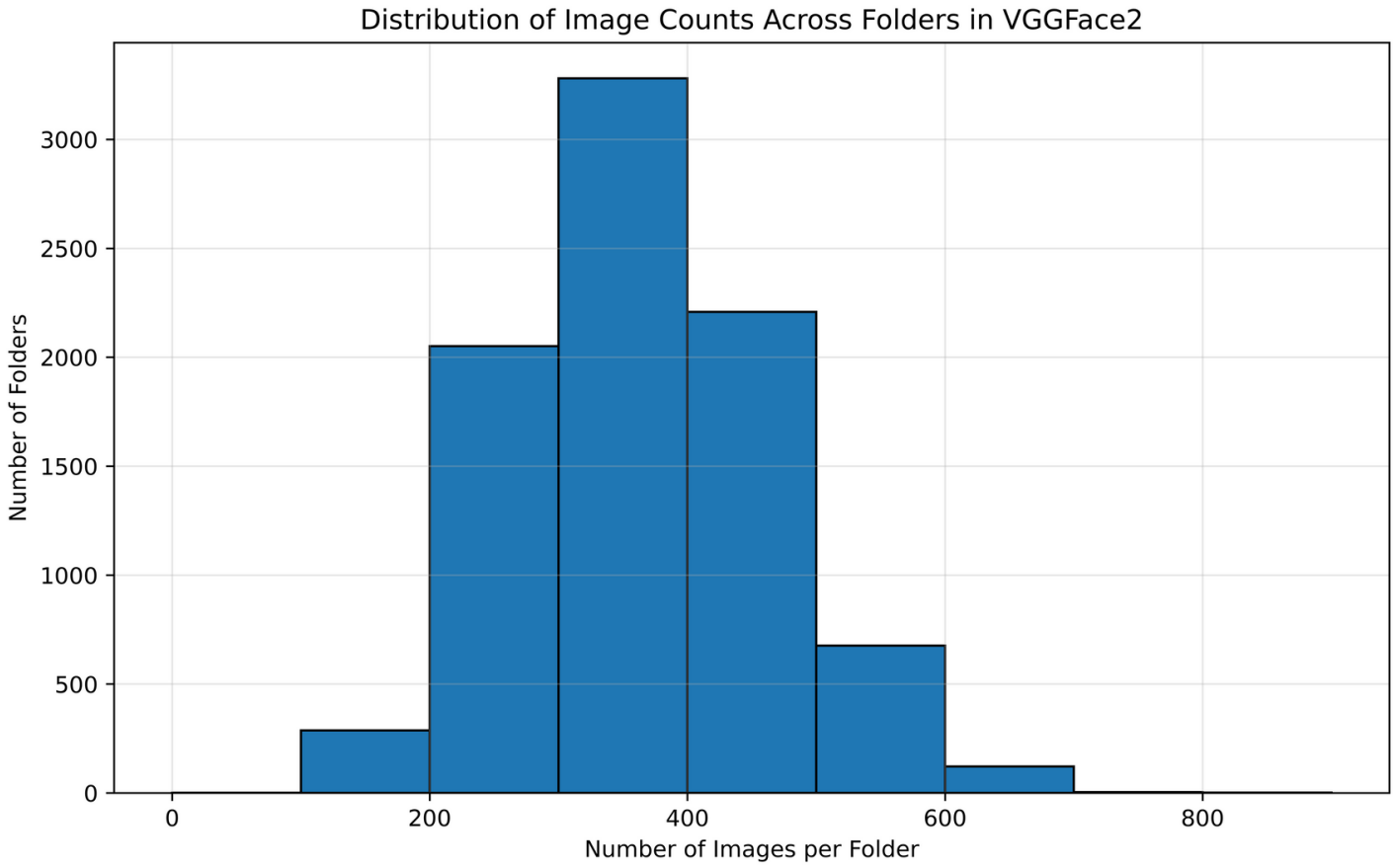
(Figure 4.2: VggFace2 Download Problem)

But we found the dataset on Kaggle

(Figure 4.3: VggFace2 Place)

After performing some Analysis on the data, we found part of data is missing, the data we found has about **3.14** million images only and **8631** identities, and here some Analysis on the data:

- Total Number of Identities: 8631
- Total Number of Images: 3141890 (3.14M)
- Min images per folder: 87
- Max images per folder: 843
- Average images per folder: 364.02

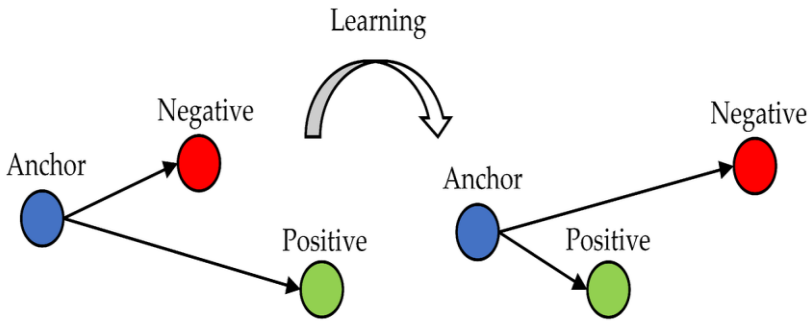


(Figure 4.4: VggFace2 Analysis)

2. Preprocessing:

For the model, we need to make some Preprocessing on the data like:

- Split
  - Train: 7767 (90%)
  - Validation: 864 (10%)
- Train
  - Detect Face (MTCNN)
  - Resize (160 x 160)
  - Augmentation
    - Horizontal Flip
    - Rotation
    - Color
  - Tensor
  - Generate Triplets (Anchor, Positive, Negative)
- Test
  - Detect Face (MTCNN)
  - Resize (160 x 160)
  - Generate Pairs



(Figure 4.5: Triplets Learning)

3. Feature Selection and Extraction:

The Data will Enter on Deep Neural Network Model, So No Need to concern about specific features

4. Model Selection and Architecture:

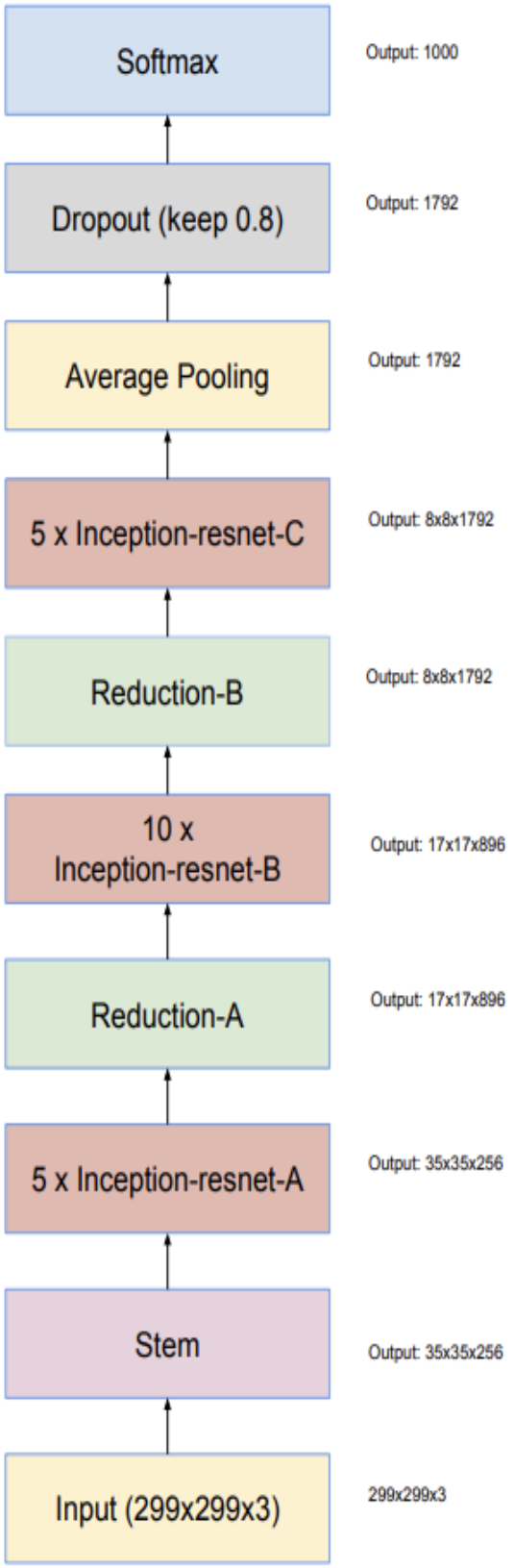
For the Model, we want a very powerful model because it is critical part in the system, so the model needs to generalize against any task, and this is not classification for this part, we need a **Recognition Model** and based on our lack of resources we decide to fine tuning model.

After reviewing most of researches in this part, we conclude to this table.

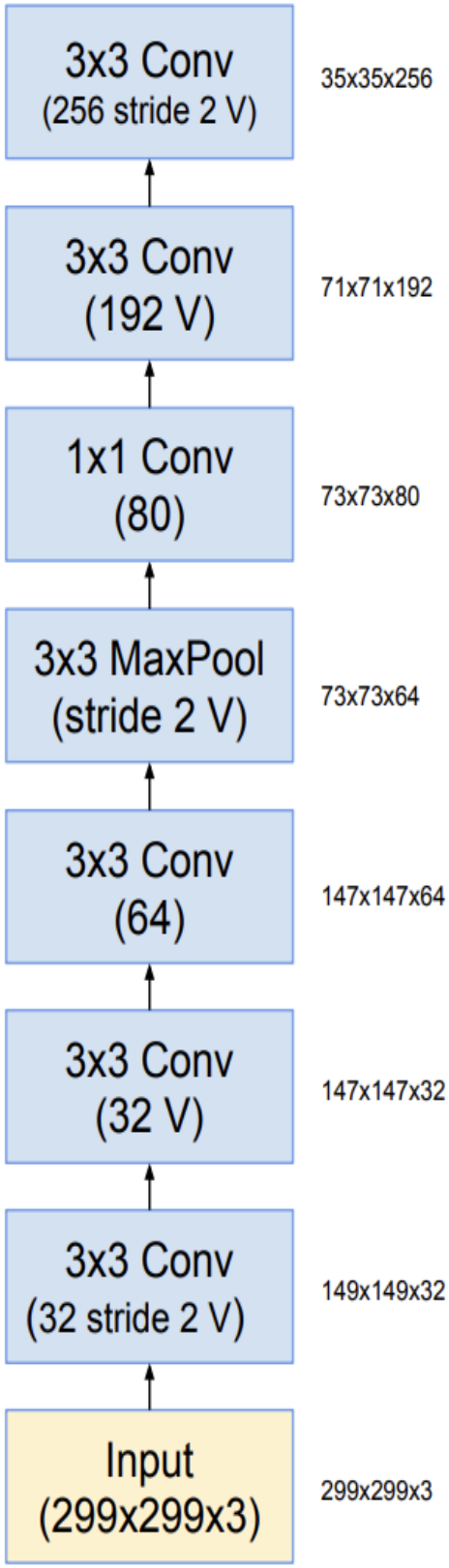
(Table 4.1: Face Verification Popular Models and Accuracy)

Model	**Accuracy**	**AUC**
Hybrid Siamese Network	98.9% (LFW)	N/A
DeepID2	99.15% (LFW)	N/A
VGG-Face	98.95% (LFW)	N/A
FaceNet (Original)	99.63% (LFW)	1.00
Improved FaceNet (EfficientNet)	99.54% (LFW)	1.00
InceptionResnetV1 (VGGFace2)	99.65% (LFW)	1.00

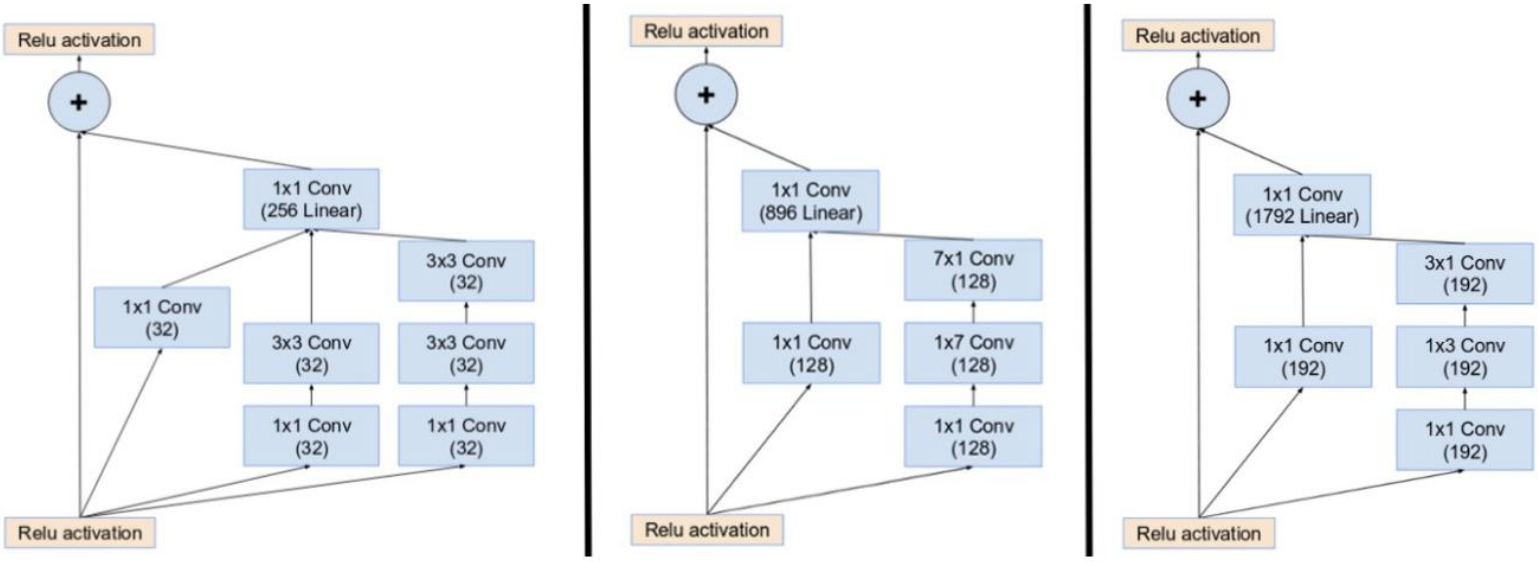
And from this table the **InceptionResnet-V1** was a good choice because it was already pretrained on VggFace2 and also we can use it to create an Embedding without deleting any layers, and it was achieve very good accuracy trained on this data and evaluate the model on LFW, Let's explores the Architecture:



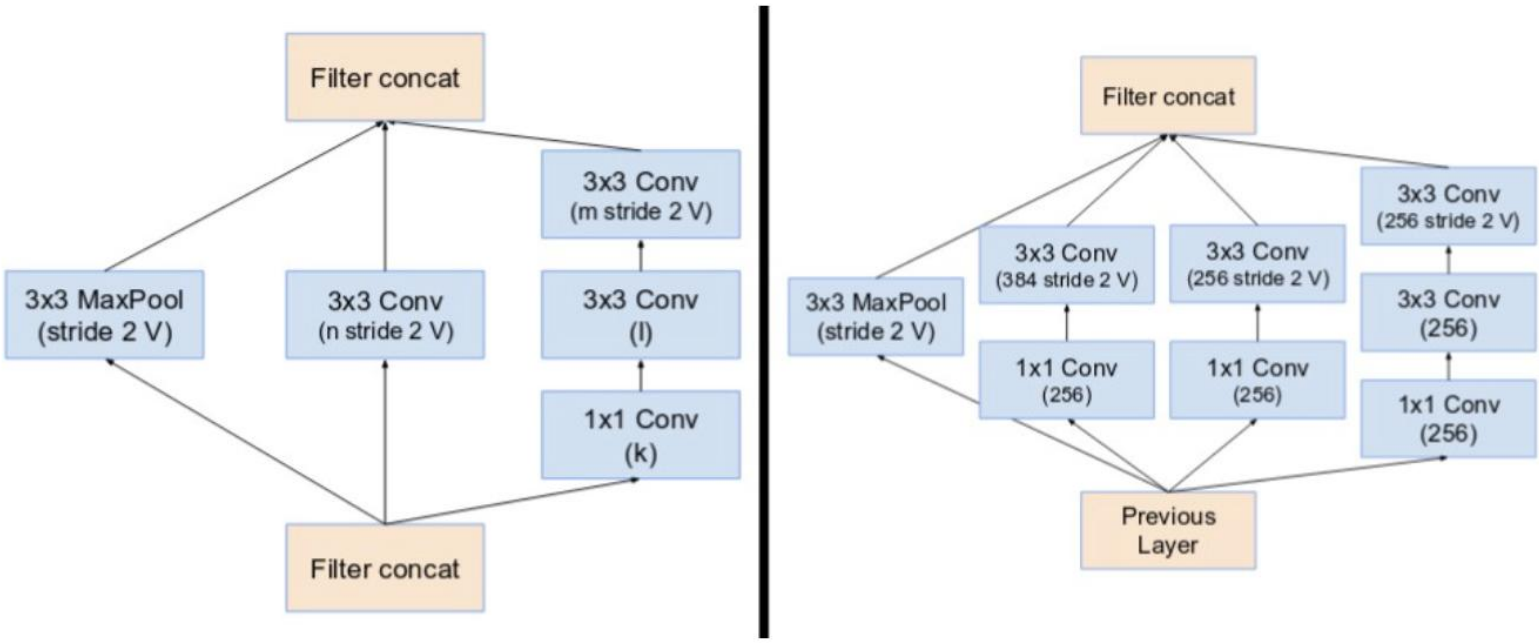
(Figure 4.6: InceptionResnet-V1) [8]



(Figure 4.7: Stem Architecture) [8]



(Figure 4.8: 3 Inception-ResNet modules: A, B and C (from left to right)) [8]

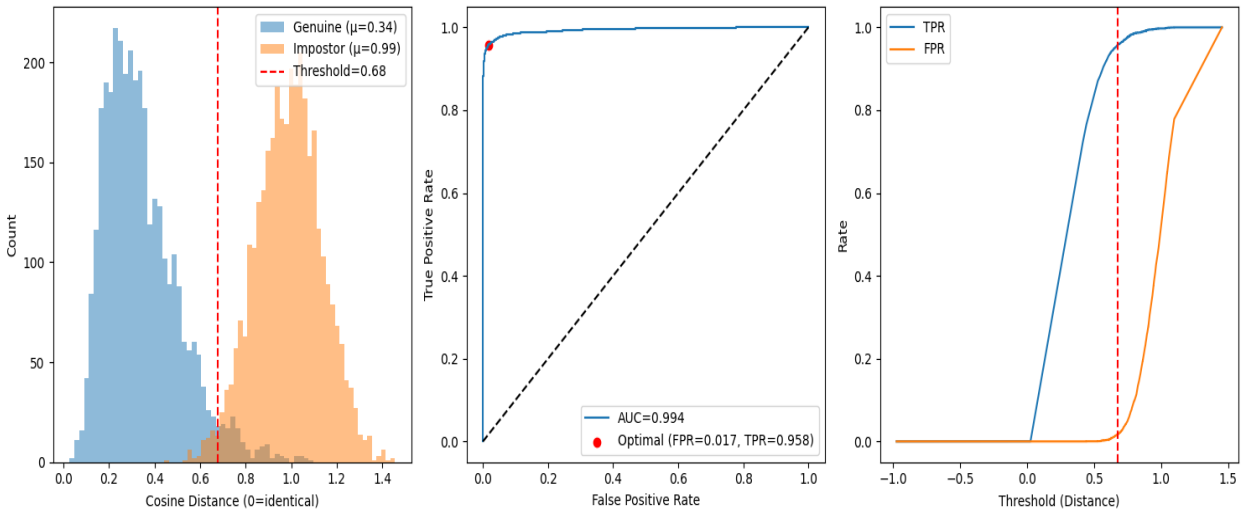


(Figure 4.9: Reduction blocks A and B (left and right)) [8]

5. Experiments and results:

We try more than model like **ResNet50** pretrained on ImageNet and it got Validation accuracy about **77%** and Test on LFW got about **62.91%** and with trying **ResNet101…**etc.

Still the same range of accuracy, but when Trying **InceptionResnet-V1** got Validation accuracy **97.09%** and Test on LFW got about **99.3 ~ 99.6%,** but when trying to fine tuning the model it get in the Validation accuracy **95.33%,** so we decided to keep the model as it is because we didn't have powerful resource to fine tuning the model with more big data.(you will find the metric below).



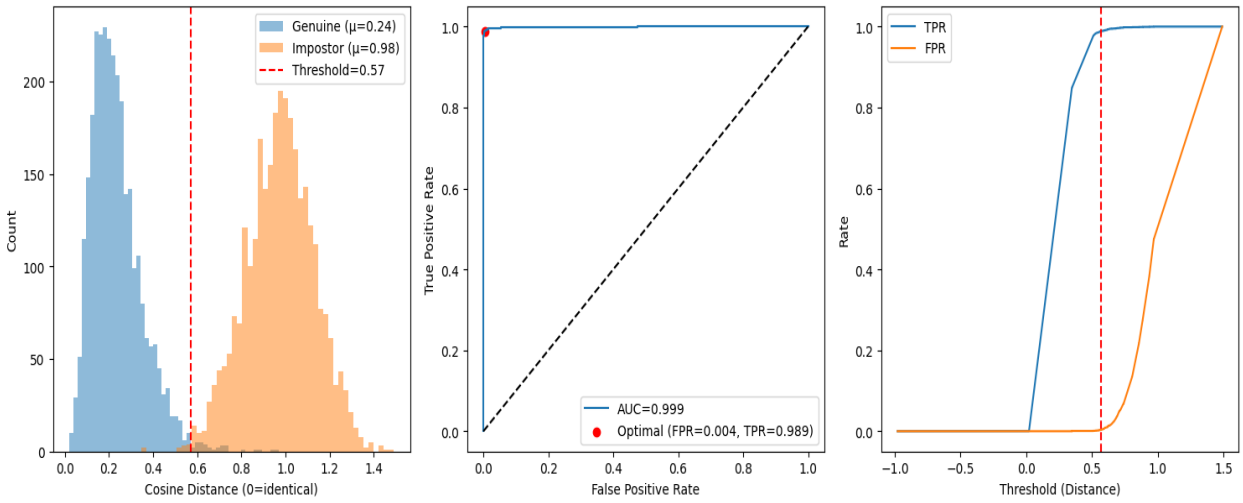
(Figure 4.10: Validation Metrics)

Validation Metrics:

- Accuracy: 97.09%
- True Positive Rate: 95.82%
- False Positive Rate: 1.65%
- Optimal Threshold: 0.6773
- AUC: 0.994

Distance Statistics:

- Genuine mean: 0.3427
- Impostor mean: 0.9889
- Minimum distance: 0.0253
- Maximum distance: 1.4549



(Figure 4.11: Testing Metrics)

Testing Metrics:

- Accuracy: 99.28%
- True Positive Rate: 98.93%
- False Positive Rate: 0.37%
- Optimal Threshold: 0.5698
- AUC: 0.999

Distance Statistics:

- Genuine mean: 0.2352
- Impostor mean: 0.9751
- Minimum distance: 0.0206
- Maximum distance: 1.4909

6. Limitation:

I Think the biggest issue here, that is not enough resources to make better adjustments but it fair enough, so the model is already 99.3~99.6%, So in real world problems this is not a problem.

4.2.2 ID Detection Model:

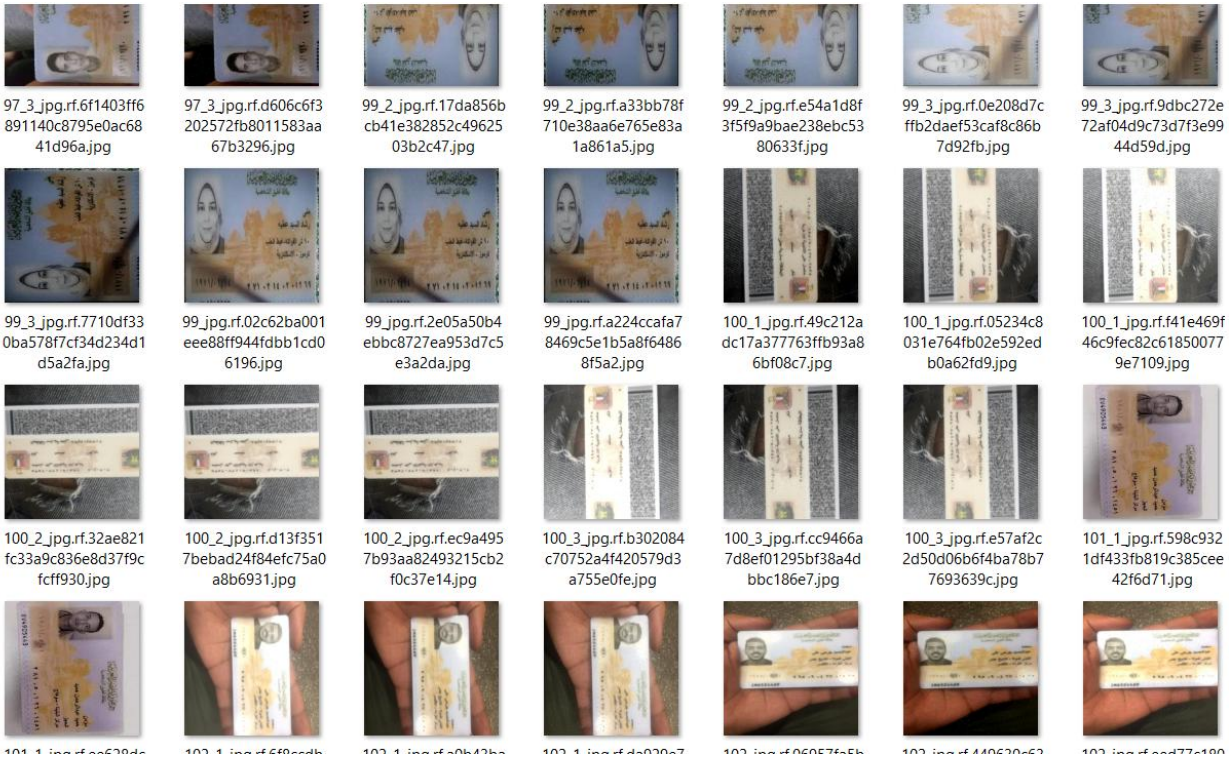
This model detects the presence and position of an ID card in a given image, even if rotated or partially obstructed.

1. **Dataset:**

To train and evaluate the two AI models in our system—**ID Detection** and **Field Detection**—we collected datasets from multiple sources to ensure diversity and robustness. These sources included publicly available image repositories, scanned and photographed ID cards from various environments, and custom image captures.

2. **Preprocessing:**

Make some augmentations like salt and paper noise , Gaussian blur and change brightness for **Generalization** (shown in Figure 4.12)



(Figure 4.12 Data Generalization)

3. Data Collection & Annotation

"We used a dataset of 1500 images from data we have been collected, manually labeled using **Roboflow** to annotate bounding boxes for ID cards in different positions (normal, rotated, upside down, etc.)."

Before training the models, the dataset was carefully prepared.

Images were collected and manually labeled using **Roboflow** to annotate bounding boxes for ID cards in different positions (normal, rotated, upside down, etc.). After annotation, the dataset was exported in **YOLOv8-compatible format**.

The dataset was structured as follows:

- **70% for training**
- **20% for validation**

- **10% for testing**

train/

├── images/

└── labels/ ← Each .txt file contains bounding boxes for fields

valid/

test/

data.yaml ← Defines class names and paths

4. ***Model Architecture & Tools***

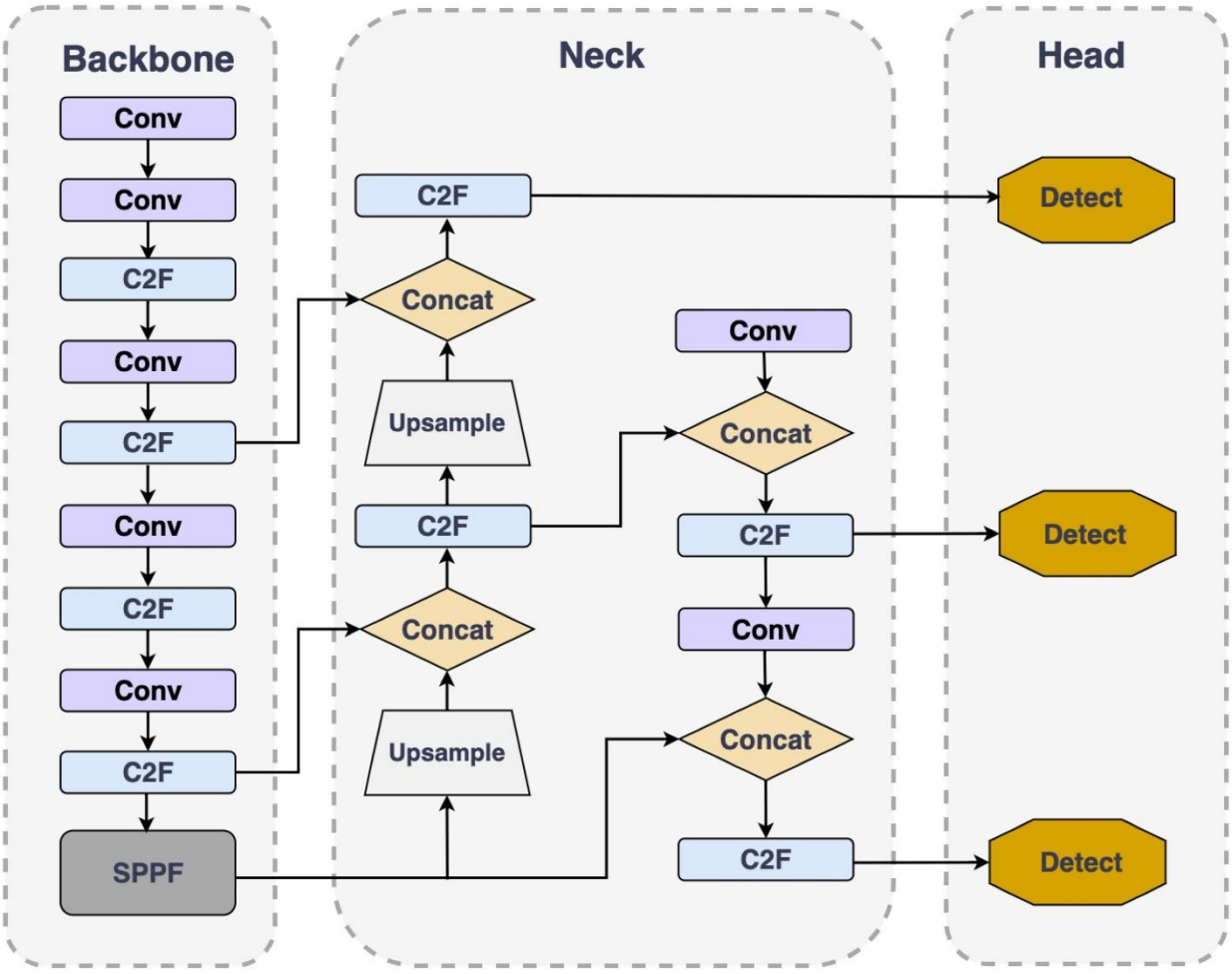
YOLOv8 – You Only Look Once Version 8

For object detection tasks such as ID card detection and field extraction, we chose **YOLOv8** (specifically, the YOLOv8n.pt variant by Ultralytics) due to its **lightweight structure, high speed, and excellent performance in real-time applications**. YOLOv8 introduces a fully updated architecture over its predecessors, offering flexibility for detection, segmentation, and classification tasks.

Why YOLOv8?

- **Fast Inference**: Suitable for front-end or real-time processing.
- **Lightweight**: The n (nano) version is optimized for low-resource environments.
- **High Accuracy**: Despite being small, it delivers competitive precision and recall scores.

- **Anchor-Free Architecture**: YOLOv8 uses anchor-free detection which simplifies training and improves generalization.
- **Easily Integrated**: Compatible with tools like Roboflow and supports PyTorch out-of-the-box.



(Figure 4.13: YOLOv8 Model Architecture)

5. Training Process

- Number of epochs : 50
- Training time : 40 mins
- Hardware used : GPU

5. Evaluation Metrics

Final Metrics at Epoch 50:

Precision: 0.8901

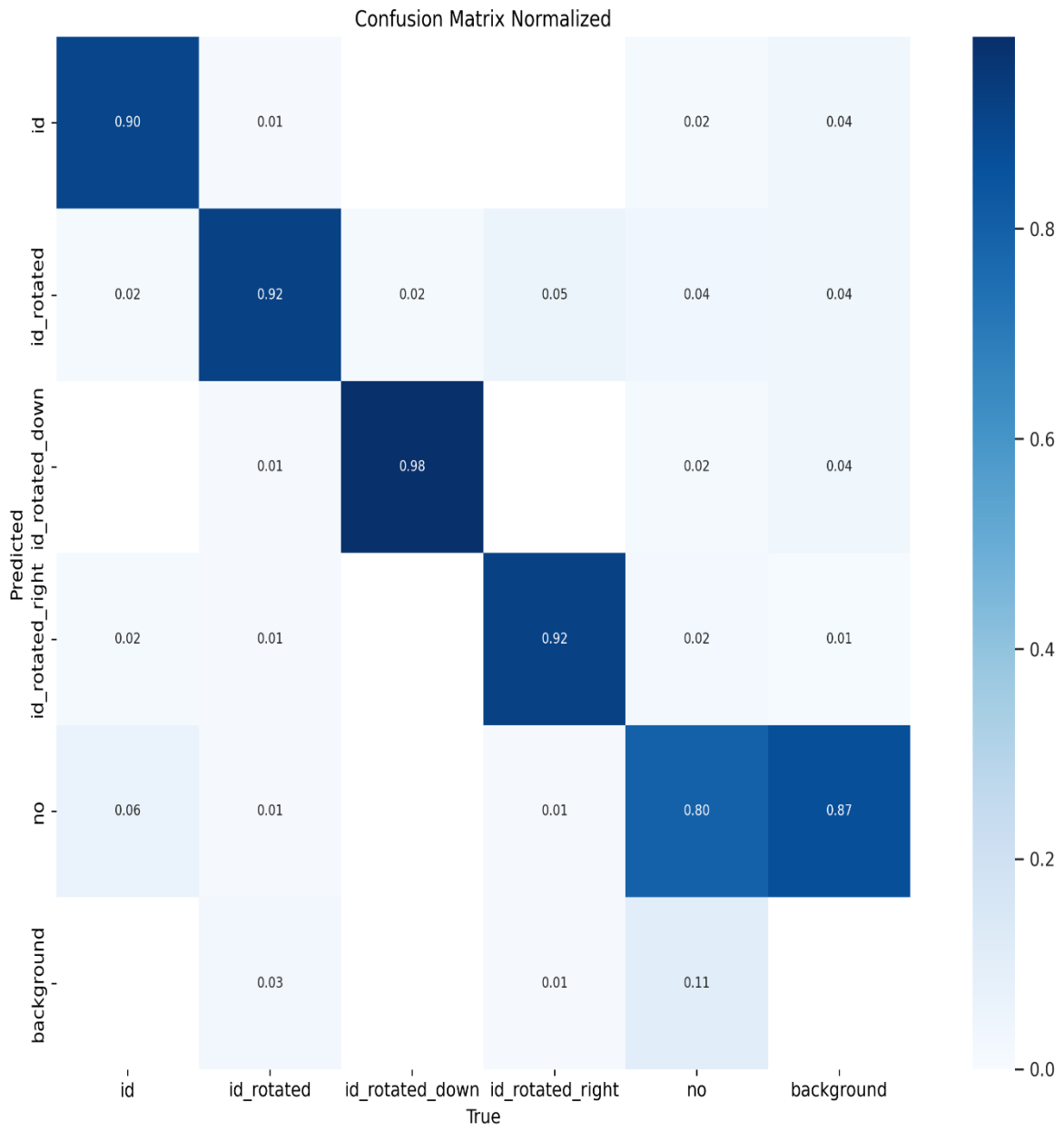
Recall: 0.88978

mAP50 (Mean Average Precision at IoU 0.5): 0.91235

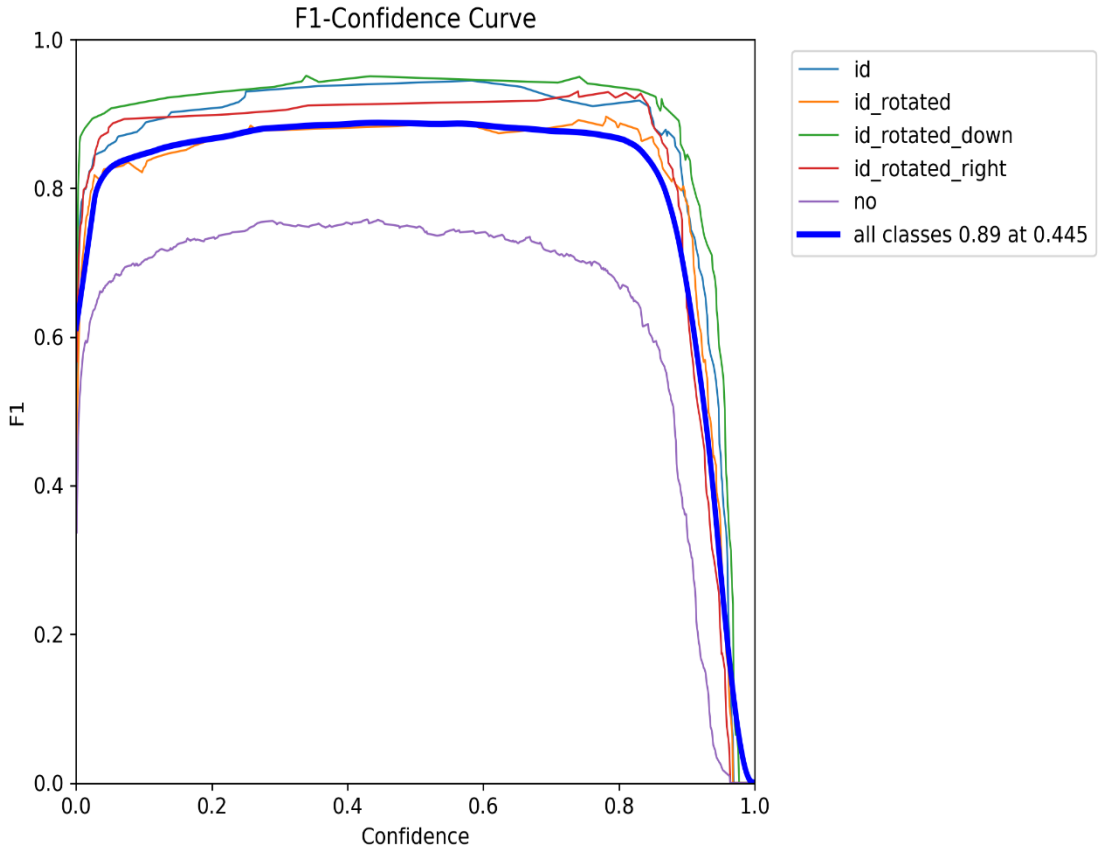
mAP50–95 (Mean Average Precision from IoU 0.5 to 0.95): 0.78725

The model showed consistent improvement and reached a strong mAP score of 0.912 at IOU 0.5

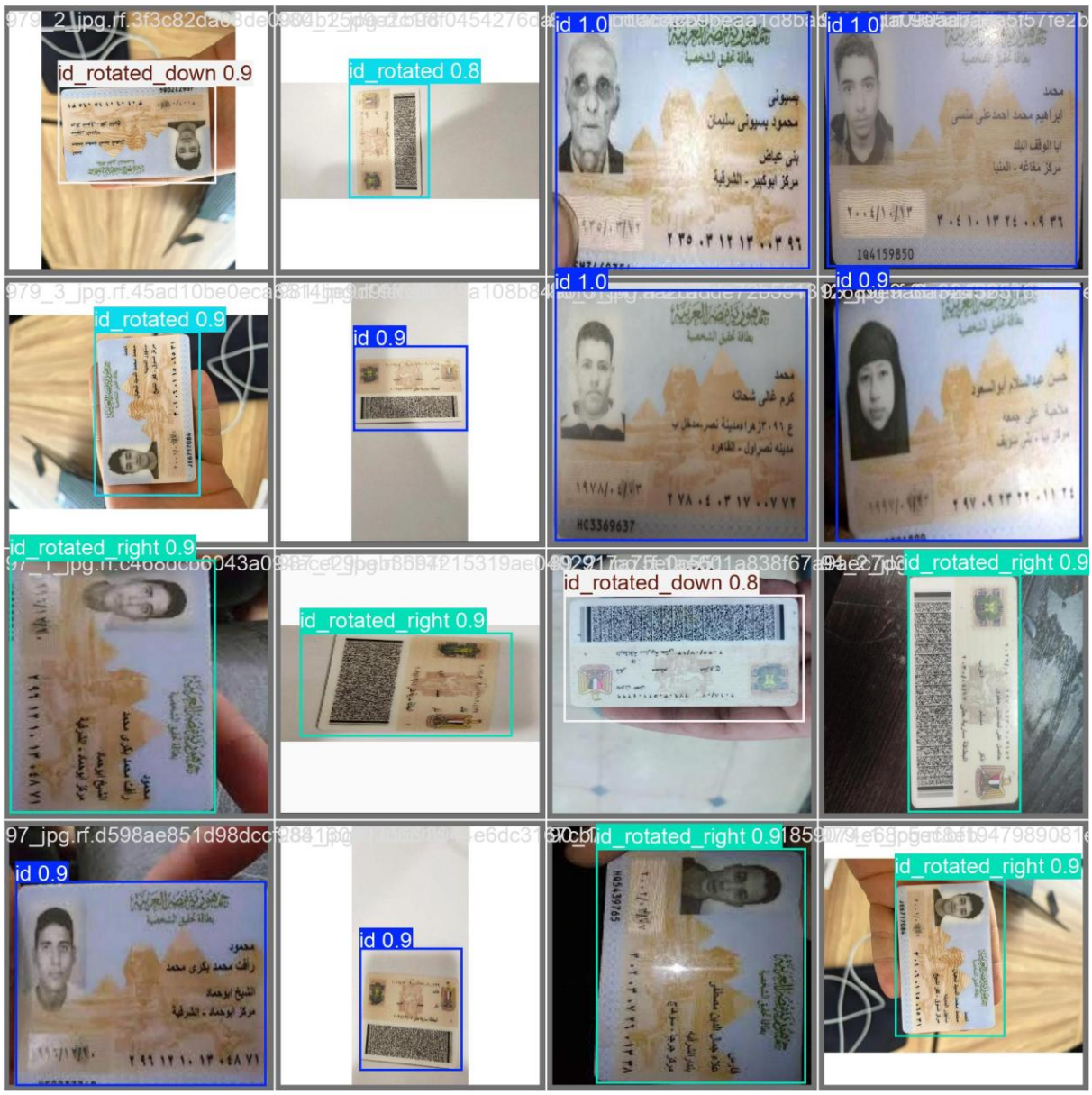
6. *Visual Results*



(Figure 4.14: ID Detection Confusion matrix)



(Figure 4.15: ID Detection F1 Curve)



(Figure 4.16: YOLOv8 Detection Predictions on Validation Set)

4.2.3 Field Detection Model:

1. Model Purpose:

After detecting and cropping the ID card from a full image using Model 1, Model 2 is responsible for identifying and localizing the individual data fields within the card. These fields include vital information such as **Full Name, National ID Number, Birth Date, Address**, and others. The goal is to extract this structured information for further processing using OCR and to support identity verification.

2. Data Collection & Labeling:

The input to this model was a **cropped ID card image**, obtained from Model 1's output. We used **Roboflow** to annotate each image manually with bounding boxes around the required fields.

Bounding Box Annotation in Roboflow:

- Each field (e.g., Name, ID Number, Address, etc.) was enclosed in a manually drawn bounding box.
- Labels were assigned to each field using a predefined list of 12 classes.
- Annotations were saved in **YOLOv8 format**, which uses normalized coordinates:

class_id, center_x, center_y, width, height

3. Model Configuration:

We used the **YOLOv8n** (nano) model for its balance between speed and accuracy, especially suitable for deployment in resource-limited environments. The model was trained separately using the labeled field detection dataset.

4. Training Process:

After training, the model achieved the following results:

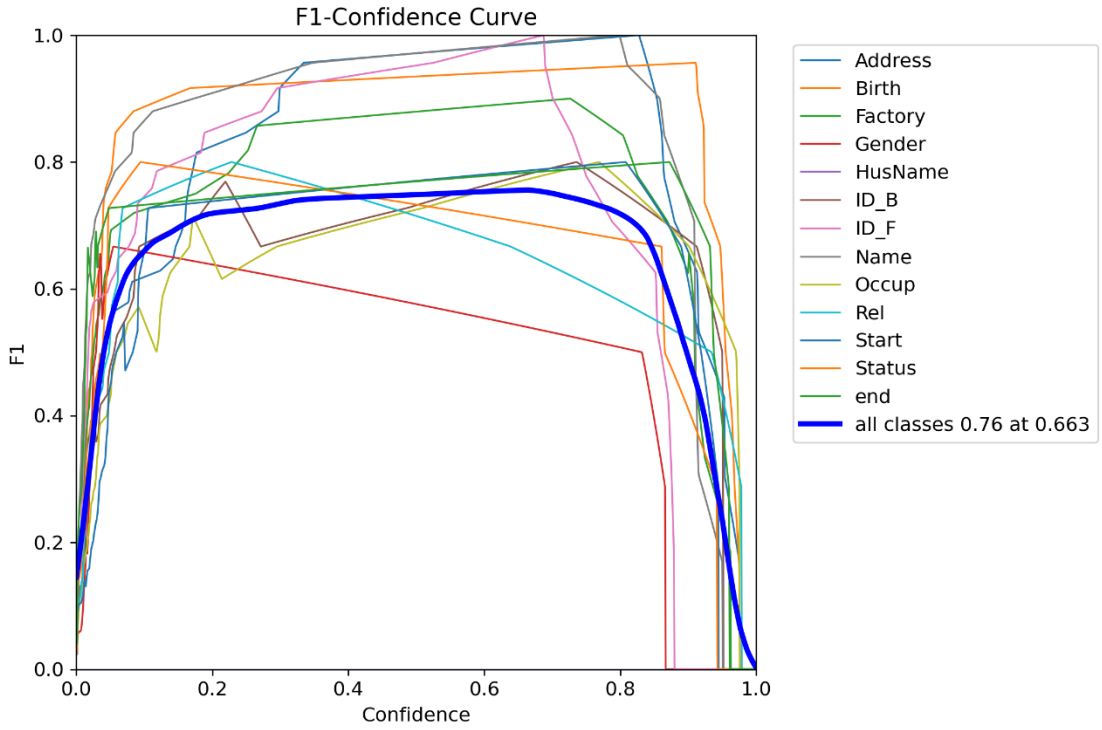
Key Metrics at Epoch 50

- **Precision (B)**: 0.96719
- **Recall (B)**: 0.67603
- **mAP50 (B)**: 0.77345
- **mAP50-95 (B)**: 0.52002
- **Total training time**: ~106 seconds

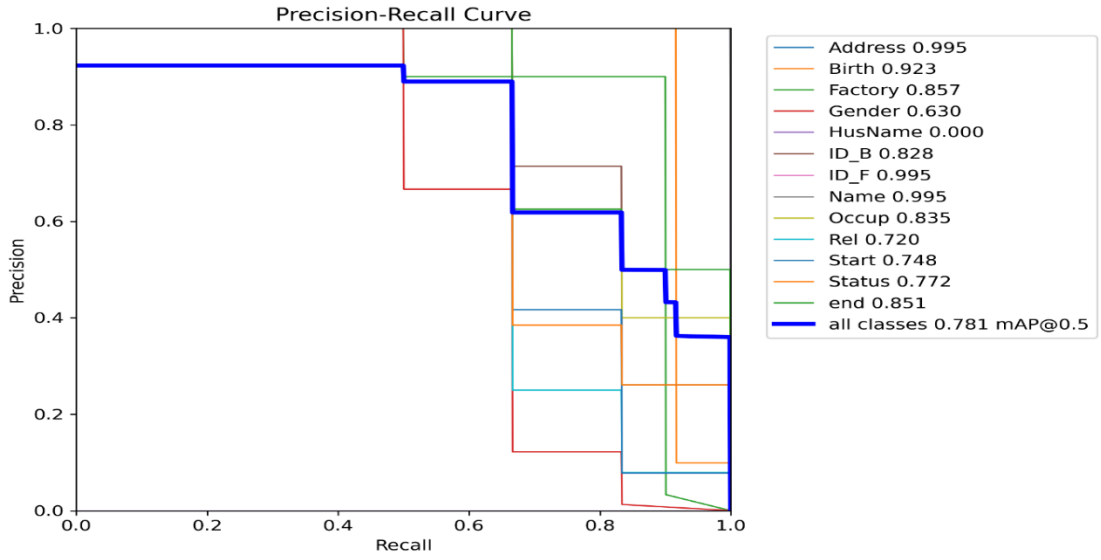
The model performed particularly well in detecting fields such as **Address** and **ID Number**, achieving up to **100% precision**. This reflects its strong ability to make accurate and reliable detections with minimal false positives.

5. Graphical Analysis

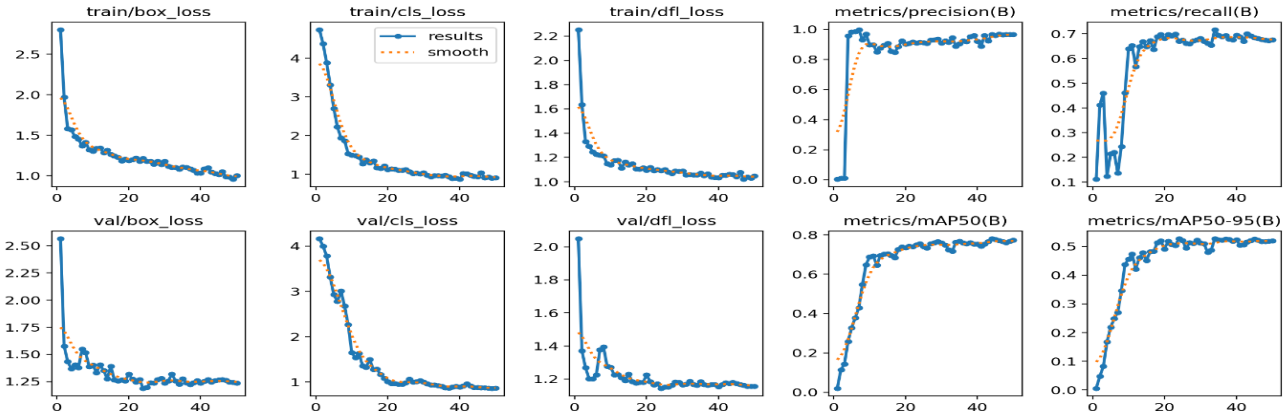
To better understand model performance across different field types, the following were generated:



(Figure 4.17: Fields Detection F1-Confidence Curve)



(Figure 4.18: Precision-Recall (PR) Curve)



(Figure 4.19: Fields Detection Results)

(Figure 4.20: YOLOv8 Detection Predictions on Validation Set)

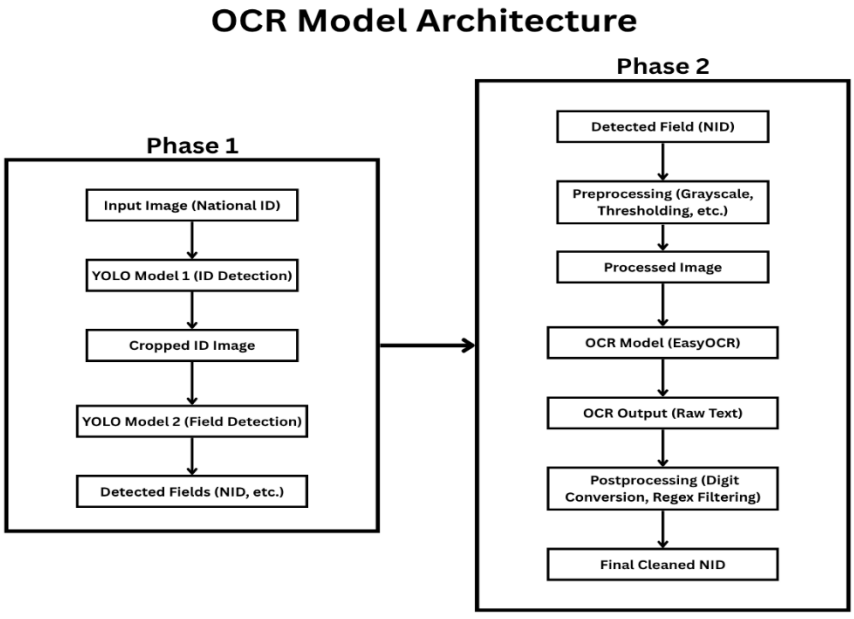
Conclusion

Together, Models 1 and 2 form a reliable AI-based pipeline for automatic ID verification. Model 2 demonstrated high precision in field-level detection, providing clean input for OCR and minimizing risk of error. These models are essential components in our secure, blockchain-based e-voting system, enabling trustworthy user registration and data extract.

4.2.4 OCR for NID Number Extraction:

1- Overview of the Pipeline

1.1-Architecture



(Figure 4.21: OCR Model Architecture)

1.2- The full pipeline consists of **two major phases**:

- **Phase 1 – Object Detection using YOLO**

To accurately isolate the NID number field from real-world images (often noisy or cluttered), we employed a **two-stage YOLO-based detection approach**:

1. **YOLO Model 1: ID Detection**
   This model locates and crops the entire National ID card from a full-scene image, such as one captured by a smartphone camera. It helps isolate the card from a background that might contain irrelevant elements.
2. **YOLO Model 2: Field Detection**
   Once the ID card is cropped, it is passed to a second YOLO model trained specifically to detect individual fields — including the National ID number, name, date of birth, gender, and more. Each detected field is then isolated for text extraction.

This two-stage detection ensures that OCR is only applied to the relevant field, improving both **speed** and **accuracy**.

- **Phase 2 –Text Extraction Using EasyOCR**

After detecting and cropping the individual NID number field, we pass it to our **OCR module** for text recognition.

Why We Chose **EasyOCR** as the OCR Engine:

We selected **EasyOCR** as the core text recognition engine in our system for several key reasons that align with the needs of extracting NID numbers from Egyptian ID cards:

- **Arabic and English Script Support**
  Egyptian ID cards use Arabic digits, which many OCR tools struggle with. EasyOCR natively supports Arabic and can accurately recognize Arabic numbers, making it ideal for our use case.
- **Deep Learning–Based and Robust**
  Unlike traditional OCR tools, EasyOCR uses deep learning (CNN + RNN + CTC), allowing it to handle real-world conditions like low image quality, skewed text, and poor lighting. This makes it reliable for text extracted from mobile-captured ID images.
- **Minimal Setup and Fast Integration**
  EasyOCR works well out-of-the-box without needing custom training, which saved time and allowed us to focus on preprocessing and postprocessing instead. This was especially helpful given our project timeline and resources.

In short, EasyOCR gave us the right balance between accuracy, language support, and ease of use for reading Arabic digits from ID cards in a real-world setting.

2- Preprocessing for OCR Accuracy

Before feeding the cropped field into EasyOCR, we apply **a series of preprocessing steps** to prepare the image for optimal recognition:

- **Grayscale Conversion**: Removes color noise and simplifies the image structure.

- **Inverse Thresholding**: Enhances the contrast between foreground text and background for better visibility.
- **Resizing**: Enlarges small text to make it more readable by the OCR model.
- **Noise Reduction**: Cleans the image to avoid false detections from background textures or artifacts.

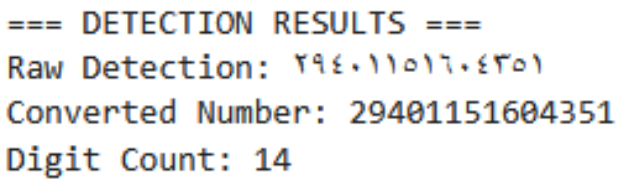
(Figure 4.22: Preprocessing result for OCR)

These steps are crucial for improving OCR accuracy, especially when dealing with **low-light photos, skewed angles, or low-resolution images** captured from mobile devices.

3- Postprocessing – Cleaning and Digit Conversion

The raw output from the OCR model is further refined through **custom postprocessing techniques**, including:

- **Arabic-to-English Digit Conversion**: Since Arabic ID cards use Arabic numerals, we map them to their English equivalents to standardize the format and allow consistent matching later.
- **Regex-Based Filtering**: We apply regular expressions to isolate the numeric content of the NID and remove unwanted characters, symbols, or spaces.
- **Validation & Storage**: The cleaned number is stored securely for use in the ID verification phase.



(Figure 4.23: OCR result sample after postprocessing)

This postprocessing ensures that the final extracted NID is **clean, accurate, and comparable** to user-provided input during the e-voting process.

4- Dataset

The model was trained and evaluated using our **custom-generated dataset** of Egyptian National ID images. These were either manually labeled or automatically cropped using the YOLO-based detection models. The dataset includes realistic variations in lighting, font, skew, and resolution to simulate real-world conditions.

5- Why OCR Is Essential

Using OCR allows us to:

- **Automate the extraction** of essential identity fields without human input.
- **Validate user identity** reliably by comparing the extracted NID against stored records.
- **Increase system security** in the e-voting process by linking users to government-issued IDs.

By integrating OCR into the pipeline, we ensure that the system remains secure, fast, and highly scalable.

Conclusion

This OCR module plays a key role in extracting the National ID Number from Egyptian ID cards. Using a YOLO-based detection pipeline and **EasyOCR**, we achieved accurate recognition of Arabic digits through targeted preprocessing and postprocessing. The approach is fast, reliable, and well-suited for real-world identity verification in our e-voting system.

4.2.5 Fake ID Detection Model

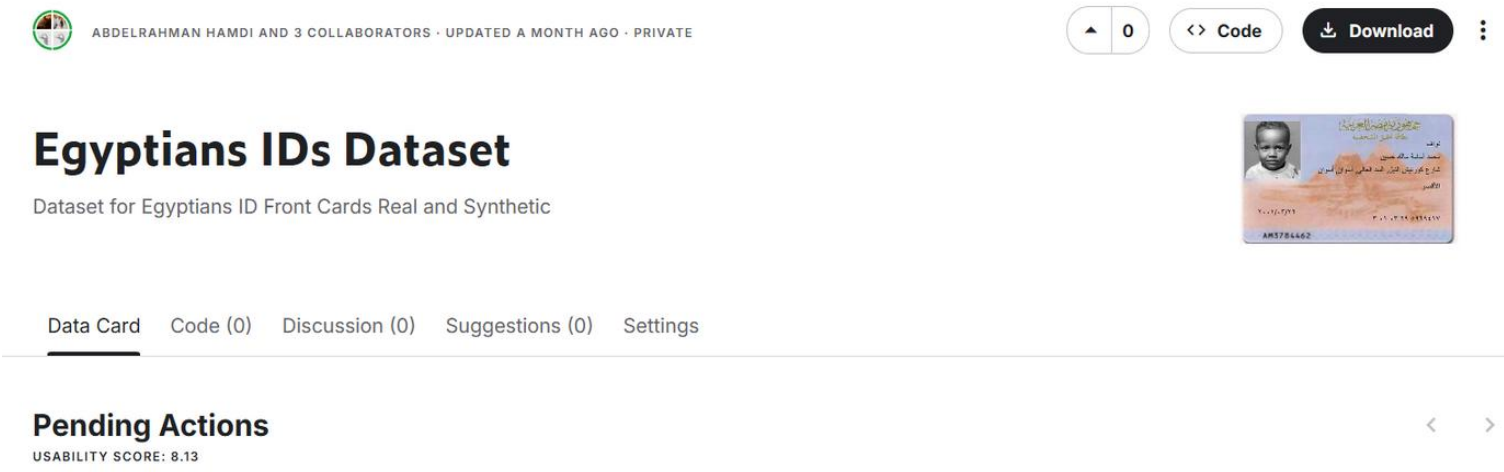
1. Dataset:

We try to search here for Egyptian ID Dataset, but we didn't find any complete dataset we can depend on it, most of the dataset we found some of duplicate images with some augmentation, so we decided to collect a dataset. Most of the dataset we found on websites like (Roboflow, google, github, facebook) specially Roboflow, it was redundant, so we collected all this dataset and images from these websites and got about **32k** image, so we start filtering step. We filter the data manually and use regex to delete images that has the same name (as images named with 14 digit of the national id number) then to make all the data clear, we make filter with CNN network like (simple CNN, MobileNetV2) to get similarity between images and decided manually if it is the same image or not.

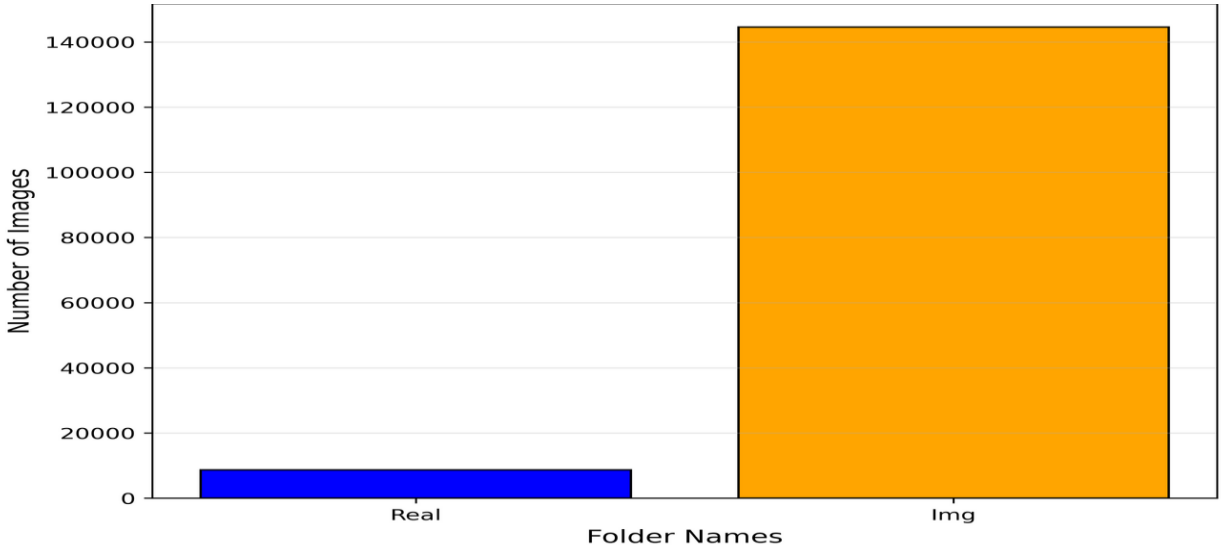
After all this, The Unique image we reach is about **8.6k** image but it is still small number of images to train robust model, so we decided to create a Synthetic Data. First we needed unique face images so we take unique face images from dataset like VggFace2 (8631 identity), CelebA (10177 identity), Synthetic Faces High Quality (SFHQ) part 4 (125K identity).

After creating we have 144k Synthetic Id image and 8.6k real image, and It is now on Kaggle

(Figure 4.24: Synthetic Id image)



(Figure 4.25: Data on Kaggle)



(Figure 4.26: Egyptian ids Analysis)

2. Preprocessing:

Preprocessing was critical due to the real-world nature of the Egyptian ID dataset. It involved the following steps:

- **Resizing:** All images were resized to a fixed dimension (128x128) to ensure uniform input size for the CNN.
- **Normalization:** Original grayscale images were scaled to the range [-1, 1] to match MobileNetV2's input expectations.
- **ELA Transformation:** Error Level Analysis (ELA) was applied by compressing the image and computing the pixel-wise difference, scaled by a factor to enhance artifacts.
- **LBP Mapping:** Local Binary Pattern (LBP) was applied to capture texture-based information. The result was normalized and converted to 3-channel format for CNN compatibility.
- **Augmentation:** During training, random flips and rotations were applied to improve generalization and robustness to orientation or mirroring.

3. Feature Selection and Extraction:

Feature extraction was **integrated** into the model pipeline rather than performed as a separate manual step. However, each input contributed distinct feature types:

- **Original Image:** Captured the raw structural content of the ID.
- **ELA Input:** Highlighted compression-based inconsistencies—useful for catching tampered regions.
- **LBP Input:** Provided local texture features to differentiate between real ID printing patterns and fake ones.

The diversity of features (high-level from CNN, low-level from LBP, and forensic from ELA) contributed to more robust classification.

4. Model Selection and Architecture:

The model uses a multi-branch hybrid CNN:

- **Original Input Branch:** Based on a frozen MobileNetV2 pretrained on ImageNet, used to extract high-level semantic features.
- **ELA Branch:** A custom CNN with two convolutional and pooling layers to learn manipulation artifacts.
- **LBP Branch:** Another custom CNN designed similarly to the ELA branch, focusing on texture.
- **Fusion Layer:** Outputs from all branches are concatenated, followed by a fully connected layer with dropout, and a final sigmoid layer for binary classification (Real/Fake).

The model was compiled with:

- **Loss:** Binary Crossentropy.
- **Metrics:** Accuracy, Precision, Recall.
- **Optimizer:** Adam with a low learning rate (3e-6).

5. Experiments and results:

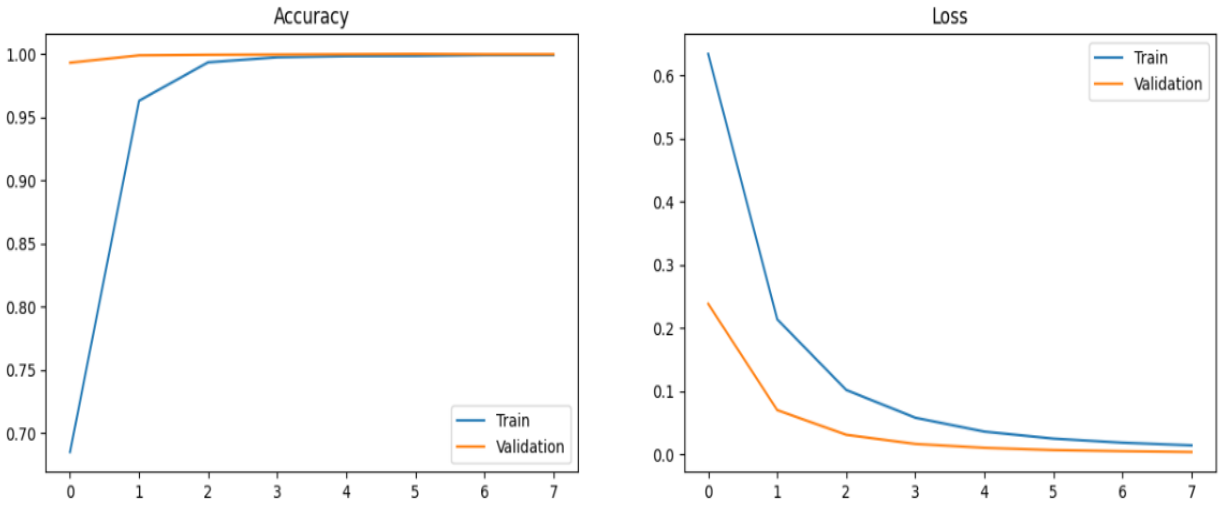
Training Setup:

- **Epochs**: 8
- **Batch Size**: 32
- **Validation Split**: 25%

- **Callbacks**: Early stopping, learning rate reduction, and CSV logging were used for stable training.

Results:

- The model achieved high training and validation **accuracy, precision, and recall**, demonstrating its ability to generalize across real and fake samples.
- Visualizations of the learning curves (accuracy, loss, precision, recall) confirmed stable convergence and strong performance across metrics.



(Figure 4.27: Fake id detection model results)

6. Limitation:

Despite promising results, several limitations were encountered:

- **Dataset Size and Balance**: The dataset was manually balanced, but the limited number of samples per class (capped at 10,000) may affect scalability.
- **Noise Sensitivity**: ELA and LBP can be sensitive to lighting, image quality, and resolution variations.

- **Overfitting Risk**: Due to the relatively shallow network branches and fixed base model (MobileNetV2), there's a risk of overfitting to specific patterns in the training data.
- **LBP Simplicity**: Standard uniform LBP was used, not more advanced variants like CN-LBP, which might extract more discriminative features.

4.3 Backend API:

The backend API for the E-Voting Blockchain system provides two main endpoints: **register** and **login**. The **register** endpoint is responsible for securely registering new users. During registration, the user submits a selfie image, a national ID image, their ID number, and a unique user address (Email). The system performs several verification steps: it detects and crops the ID card from the provided image, checks the authenticity of the ID card to ensure it is not fake, extracts the ID number field using OCR, and compares the extracted ID number with the one provided by the user. Additionally, it generates facial embeddings from both the selfie and the ID card image and compares them to confirm that the same person appears in both and check this ID number is the first time registered on the blockchain. If all checks pass, the user is successfully registered.
The **login** endpoint handles user authentication. Here, the user submits a selfie image and their user address. The system generates facial embedding from the provided selfie and compares it to the stored embedding associated with the user address (retrieved from blockchain). If the similarity score meets the required threshold, access is granted; otherwise, access is denied.

Both endpoints are designed to ensure the integrity and security of the voting process by leveraging advanced face recognition and document verification techniques.

4.4 Blockchain:

1. System Design and Architecture

1.1 Architectural Overview

The system implements a layered, modular architecture designed around the Facade pattern with clear separation of concerns.

The platform uses zkSync's Account Abstraction feature to hide the complexity of blockchain public/private key management from users, making the system more user-friendly and accessible, especially for those unfamiliar with blockchain technology.

1.1.1 Architectural Layers

- **Presentation Layer:** VotingFacade.sol - Unified interface for all user interactions
- **Logic Layer:** Specialized components handling domain-specific responsibilities
- **Data Access Layer:** State management and storage abstractions
- **Security Layer:** Access control and validation mechanisms

1.2 Component Architecture

The system consists of six primary components, each implementing specific domain responsibilities:

1.2.1 VotingFacade Component

Purpose: Central orchestrator providing unified system interface

Responsibilities:

- Proposal lifecycle management coordination
- Vote processing and validation coordination
- User authentication and authorization
- Administrative function delegation
- Cross-component communication facilitation

1.2.2 Access Control Framework

AccessControlManager: Implements comprehensive role-based access control with:

- **Administrative Role Management:** System-wide administrative capabilities
- **Voter Verification System:** Multi-tier voter authentication
- **Dynamic Permission Management:** Runtime role assignment and revocation
- **Authorization Validation:** Consistent permission enforcement across components

AccessControlWrapper: Provides reusable security abstractions and modifier patterns for consistent access control implementation across the system.

1.2.3 Proposal Management Framework

ProposalOrchestrator: Central coordination hub for proposal operations:

- **Proposal Lifecycle Coordination:** Manages creation, voting, and finalization processes

- **Cross-Component Integration:** Coordinates validation, state, and tallying systems
- **Logic Implementation:** Implements complex proposal management workflows
- **Data Aggregation Services:** Provides comprehensive proposal information retrieval

ProposalState: State management and lifecycle control system:

- **Status Tracking:** Manages proposal states (PENDING, ACTIVE, CLOSED, FINALIZED)
- **Temporal Management:** Time-based automatic status transitions
- **Participation Management:** Voter participation tracking and validation
- **Configuration Management:** Voting option and mutability control

1.2.4 Validation Framework

ProposalValidator: Comprehensive validation system ensuring:

- **Input Validation:** Parameter verification for all operations
- **Business Rule Enforcement:** Compliance with voting regulations and constraints
- **Temporal Constraint Validation:** Time-based operation validation
- **State Consistency Checks:** Ensuring operations match current system state

1.2.5 Voter Management System

VoterRegistry: Complete voter lifecycle management:

- **Registration Services:** Secure voter enrollment with identity verification

- **Participation Analytics:** Historical engagement and activity tracking
- **Privacy Protection:** Secure handling of sensitive voter information
- **Integration Services:** Seamless integration with access control systems

2. Implementation

2.1 Smart Contract Development

The implementation utilizes Solidity 0.8.23, leveraging the latest security features and gas optimization techniques. The development process followed a test-driven development approach, ensuring comprehensive coverage and reliability.

2.1.1 Key Implementation Features

- **Immutable Contract References:** All component addresses are set as immutable during deployment, preventing unauthorized modifications
- **Interface-Based Architecture:** All components implement well-defined interfaces, promoting loose coupling and testability
- **Event-Driven Communication:** Comprehensive event logging for transparency and off-chain monitoring
- **Gas-Optimized Operations:** Strategic use of storage patterns and computation optimization

3. Security Framework

3.1 Multi-Layered Security Architecture

The system implements a comprehensive security framework based on defense-in-depth principles:

3.1.1 Access Control Layer

- **Role-Based Permissions:** Hierarchical access control with admin and verified voter roles
- **Dynamic Authorization:** Runtime permission validation for all operations
- **Principle of Least Privilege:** Minimal necessary permissions for each operation

3.1.2 Validation Layer

- **Input Sanitization:** Comprehensive validation of all user inputs
- **Logic Validation:** Enforcement of voting rules and constraints
- **State Consistency Checks:** Ensuring operations are valid for current system state
- **Temporal Validation:** Time-based constraint enforcement

3.1.3 Data Integrity Layer

- **Immutable Storage:** Blockchain-based immutable record keeping
- **Cryptographic Verification:** Hash-based data integrity verification
- **Audit Trail:** Complete transaction history for all operations

3.2 Privacy and Transparency Balance

- **Selective Information Disclosure:** Authorized access to detailed proposal information
- **Vote Privacy:** Individual vote protection while maintaining system transparency
- **Participation Tracking:** Secure voter engagement monitoring
- **Public Verifiability:** Open verification of vote tallies and results

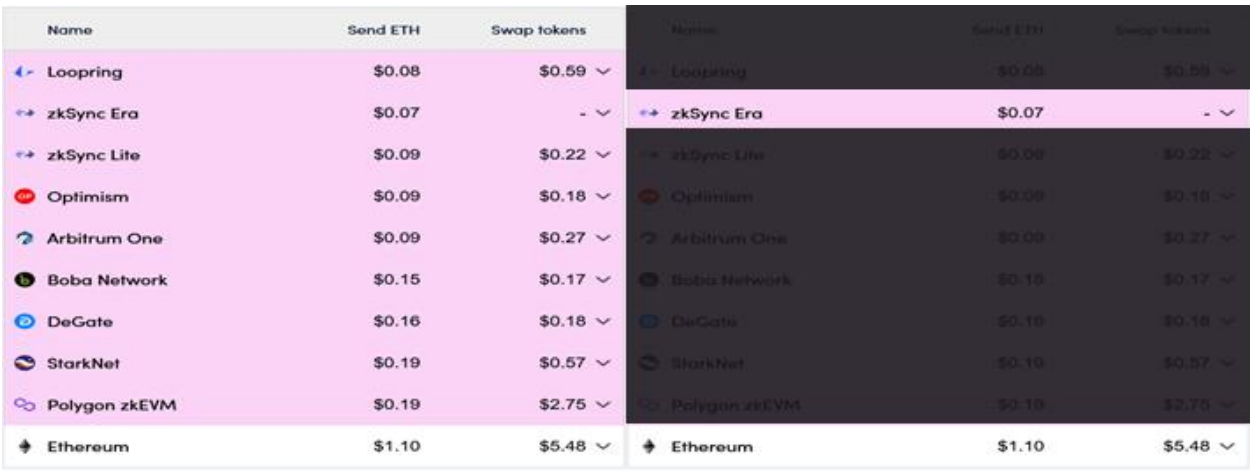
4. Deployment

After the survey, we found that Ethereum has common issues like high-cost fees, lower transaction rate per second and long Deployment and confirmation time.

And all this problem was solved with **Zk-Sync** blockchain that is Ethereum Layer 2 solution that has reduced the cost with about 95-98% and with about ~2000 transaction per second unlike Ethereum that has 15-30 transaction per second and this can showed in small test on the two blockchain:

(Table 4.2: ZK-Sync vs Ethereum)

Metric	**ZK-Sync**	**Ethereum L1**
Deployment Time	~2 seconds	~20 seconds
Tx. Confirmation Time	~0-1 minutes	~3-9 minutes
Deployment Cost (USD)	$6.19	$11.759
Avg Transaction Cost (USD)	$0.061	$0.891
Transaction Throughput (Transactions per second)	~2,000 TPS	~15-30 TPS



Name	Send ETH	Swap tokens
Loopring	$0.08	$0.59
zkSync Era	$0.07	-
zkSync Lite	$0.09	$0.22
Optimism	$0.09	$0.18
Arbitrum One	$0.09	$0.27
Boba Network	$0.15	$0.17
DeGate	$0.16	$0.18
StarkNet	$0.19	$0.57
Polygon zkEVM	$0.19	$2.75
Ethereum	$1.10	$5.48

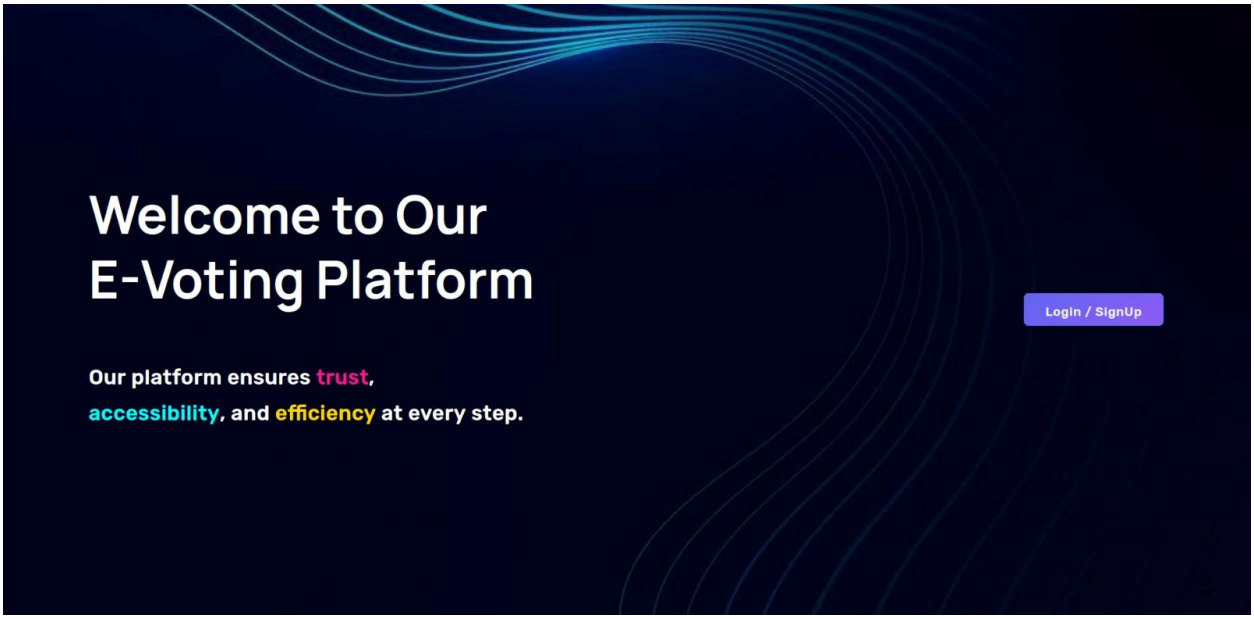
(Figure 4.28: ZK-Sync vs Ethereum Cost)

Chapter Five:

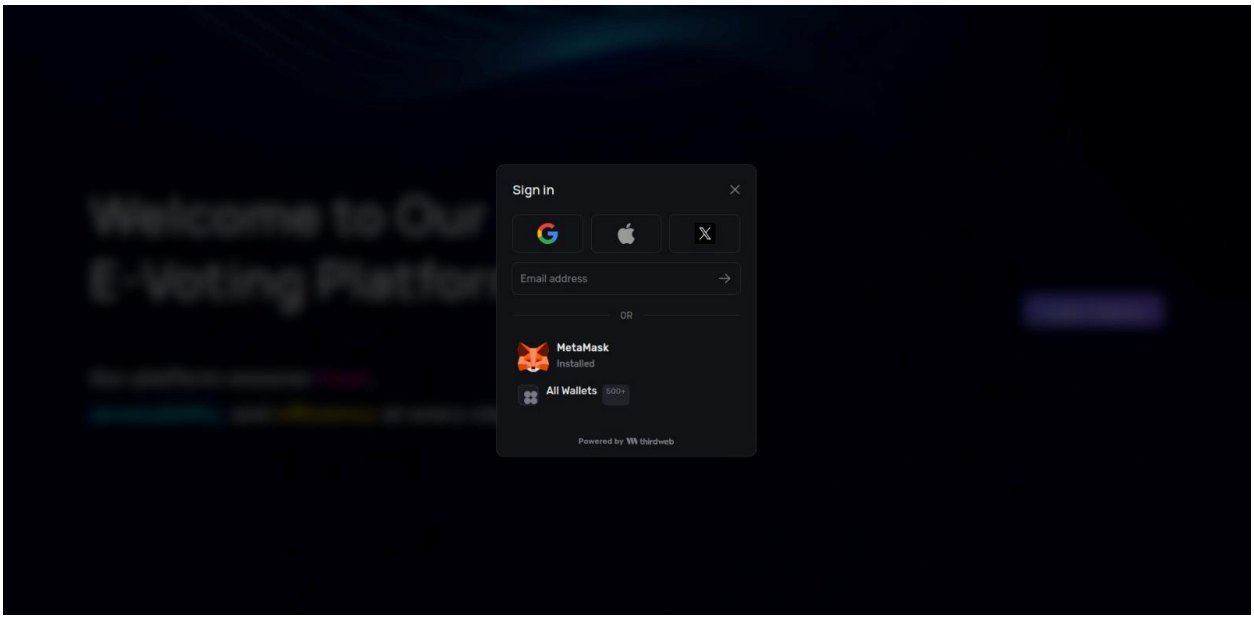
Run the Application

Chapter 5: Run the Application:

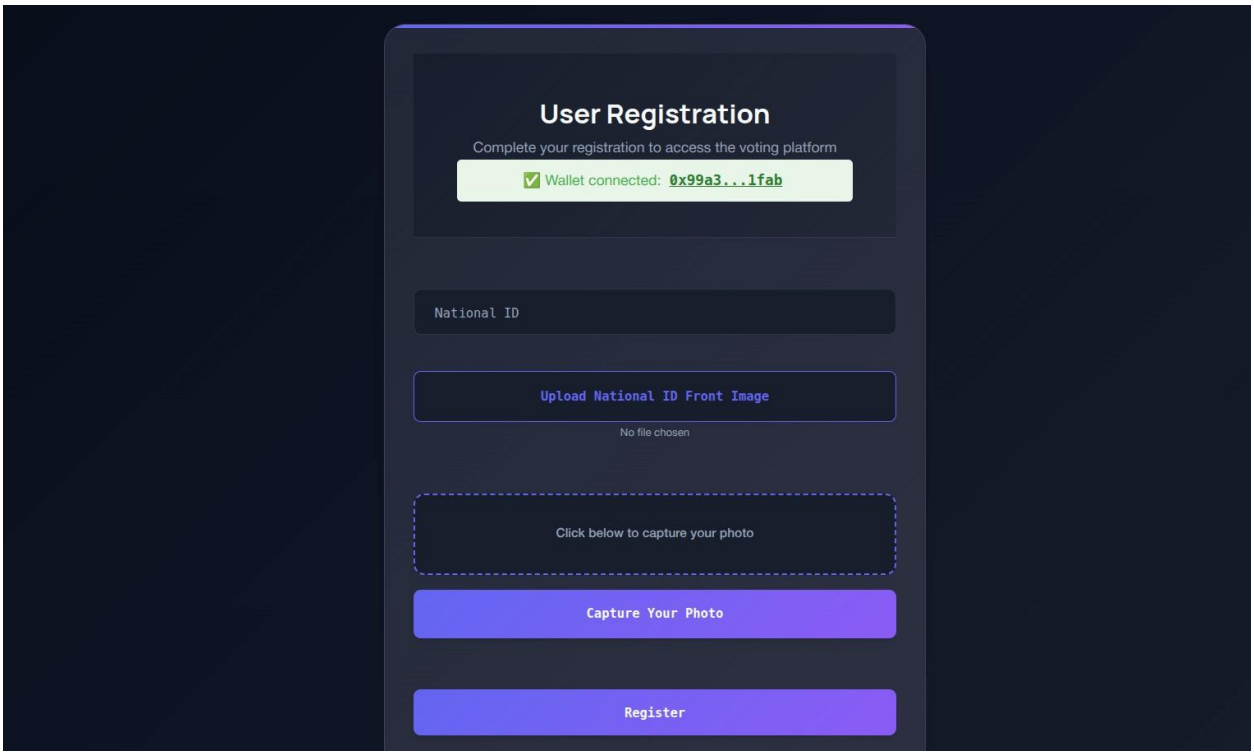
5.1 Frontend Application:



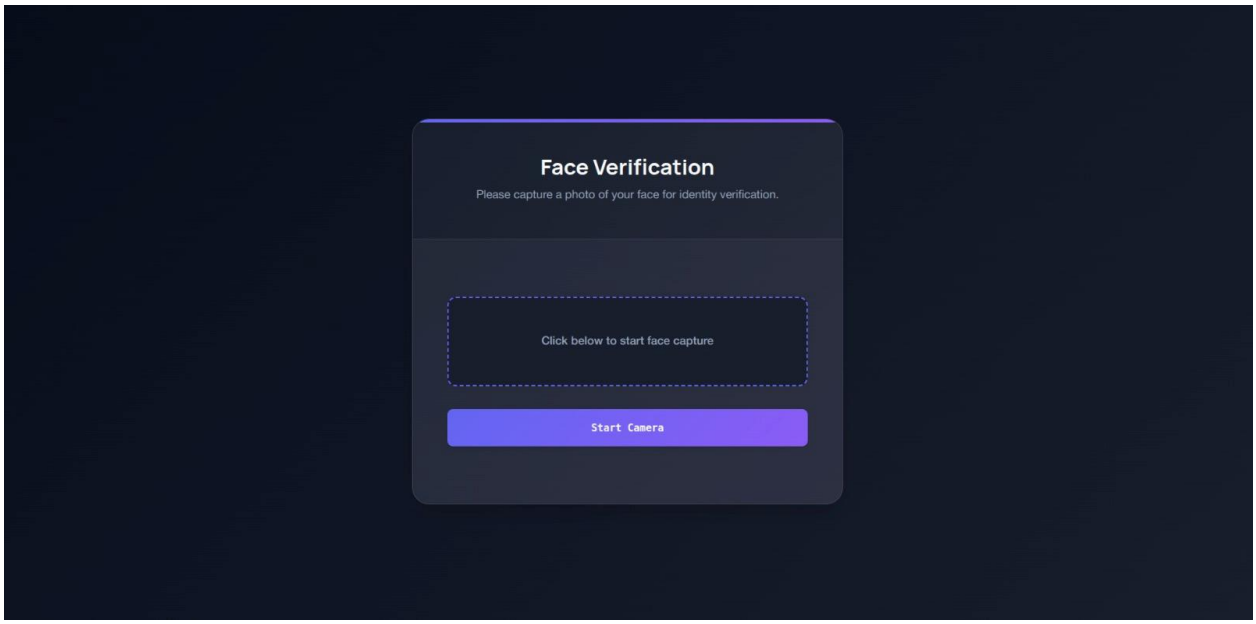
(Figure 5.1: Home Page Screen)



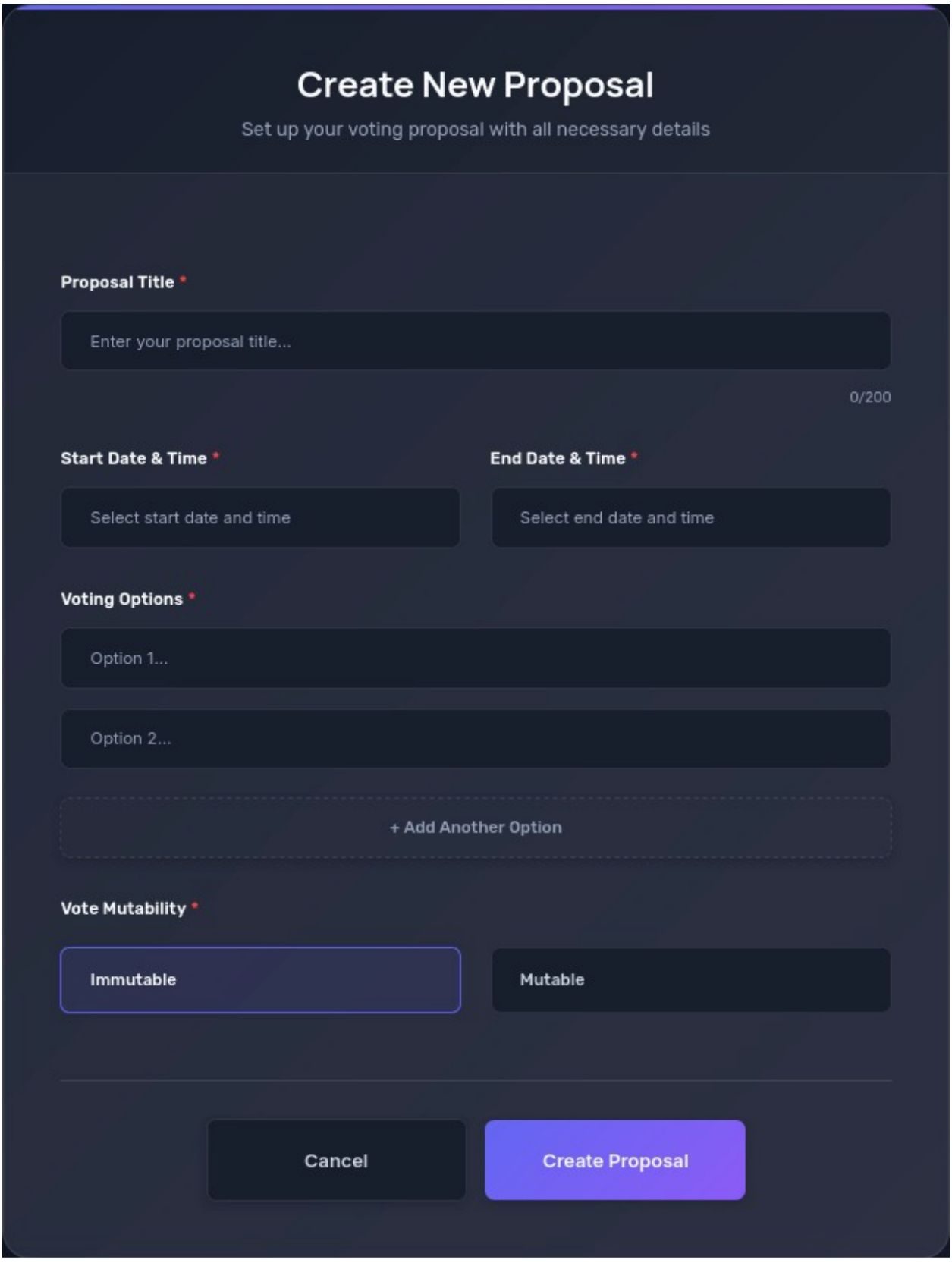
(Figure 5.2: Login and Registration screen)



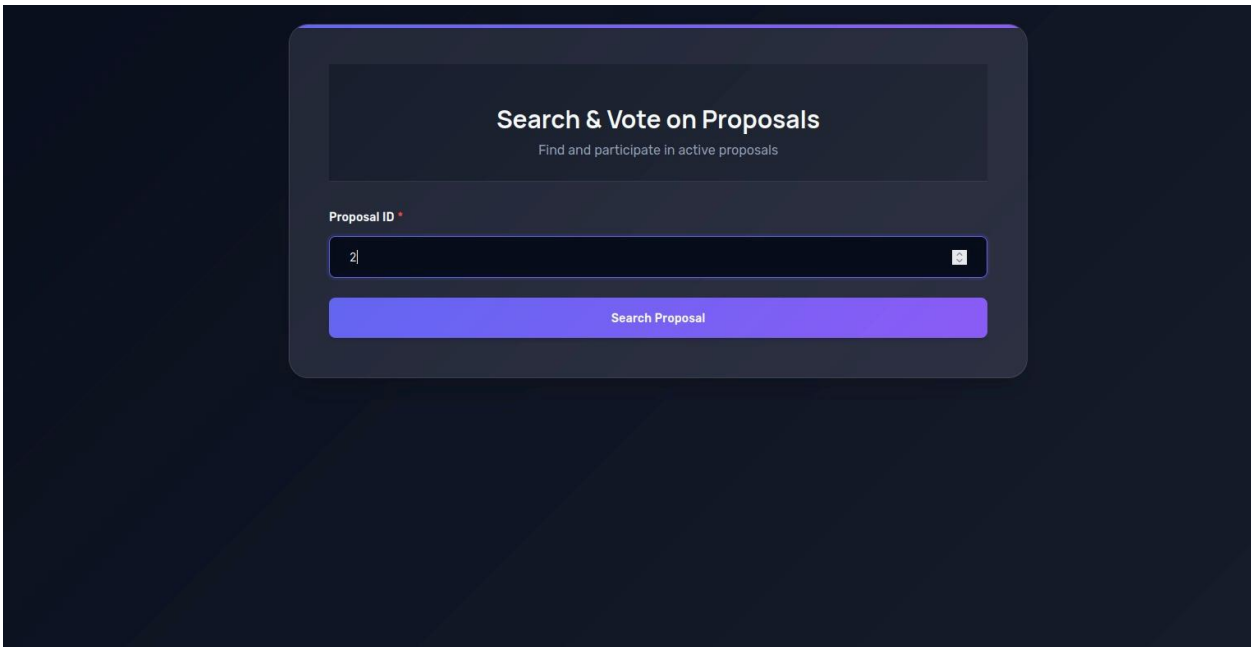
(Figure 5.3: Register screen)



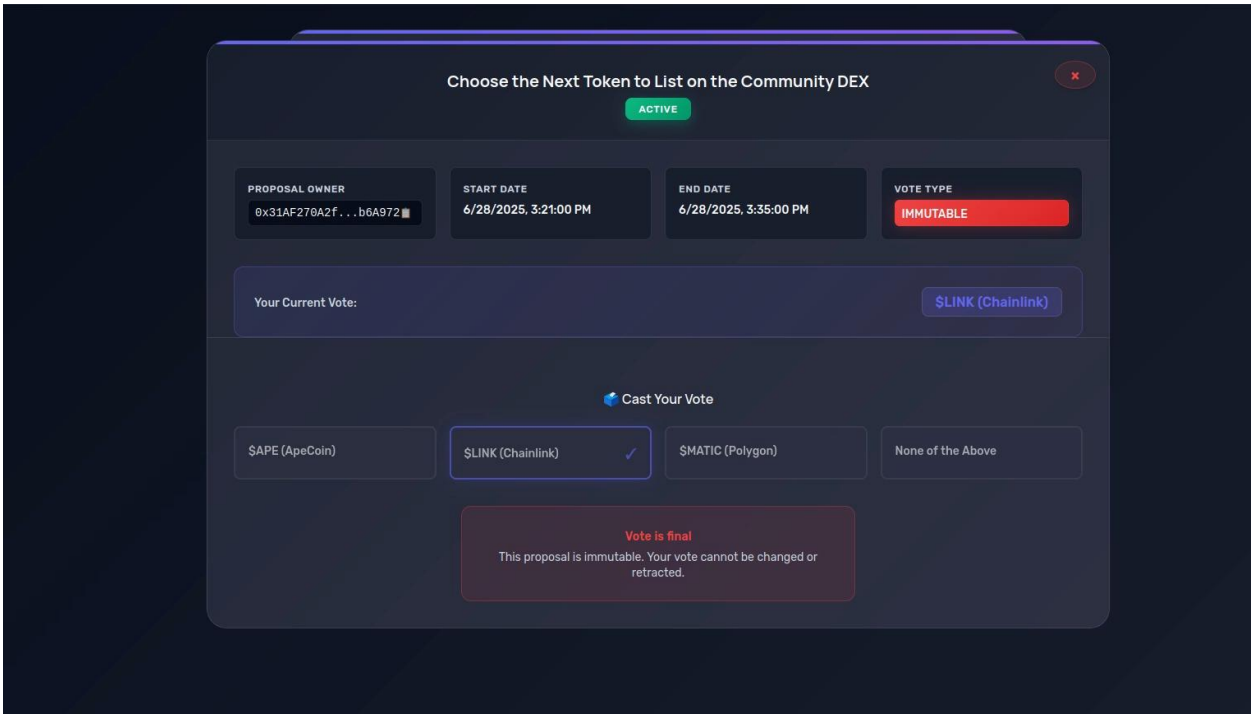
(Figure 5.4: Login Face Verification)



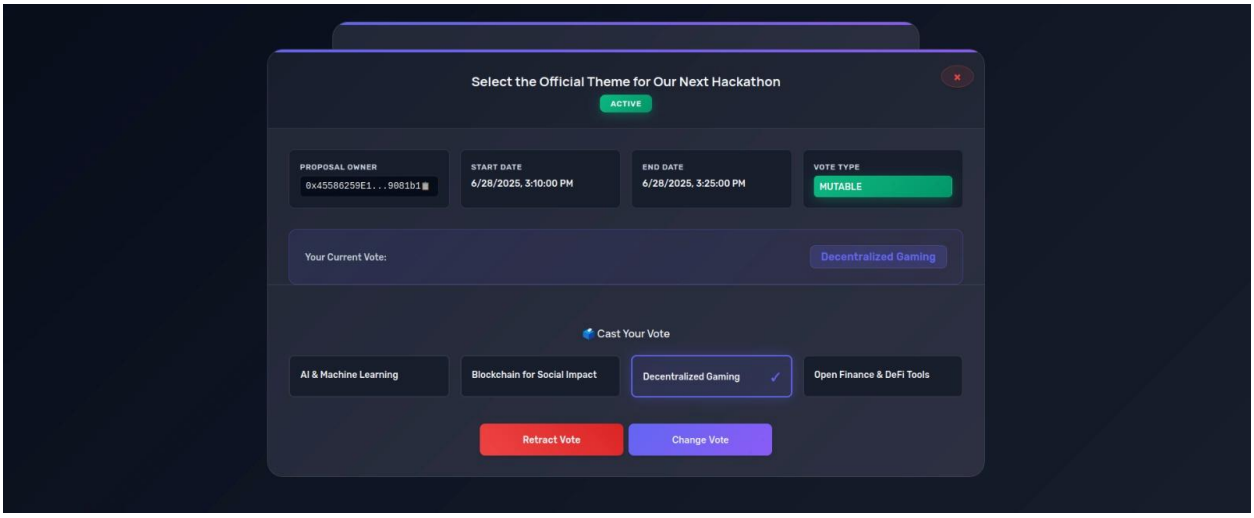
(Figure 5.5: Create New Proposal)



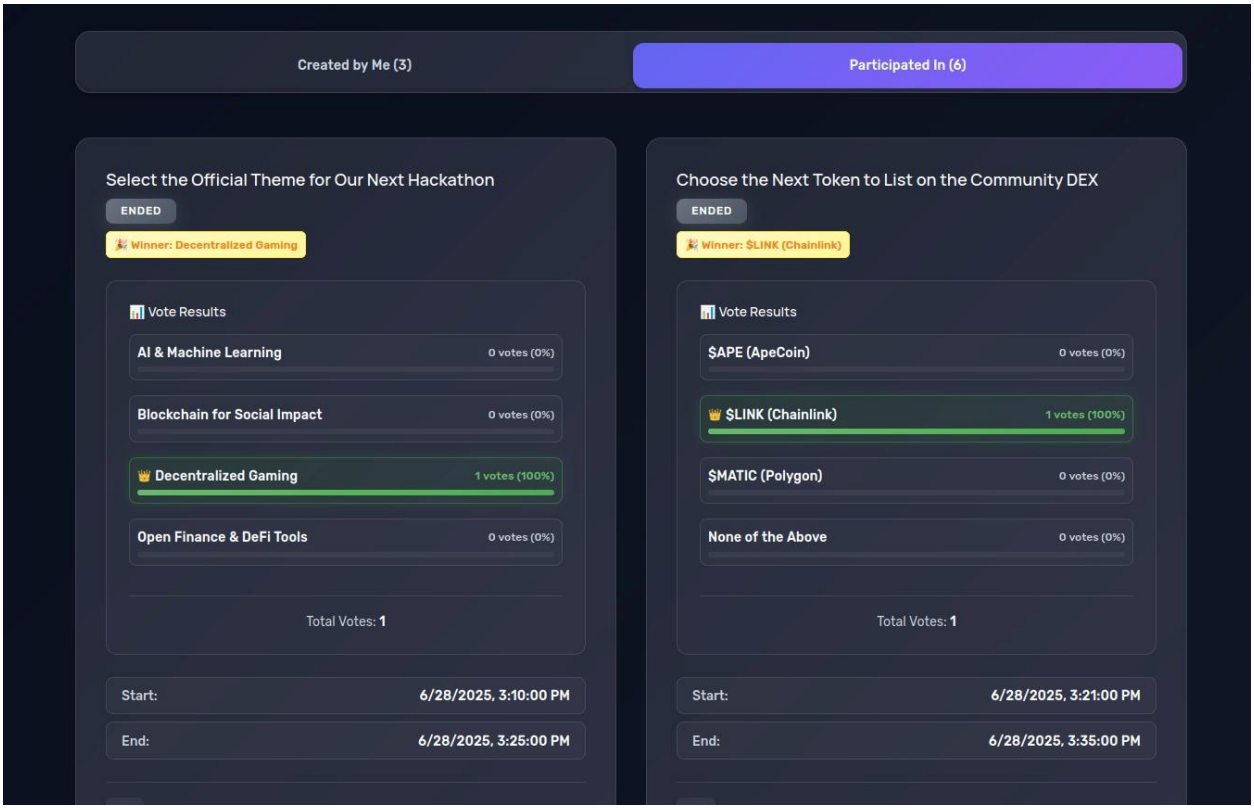
(Figure 5.6: Search & Vote on Proposal)



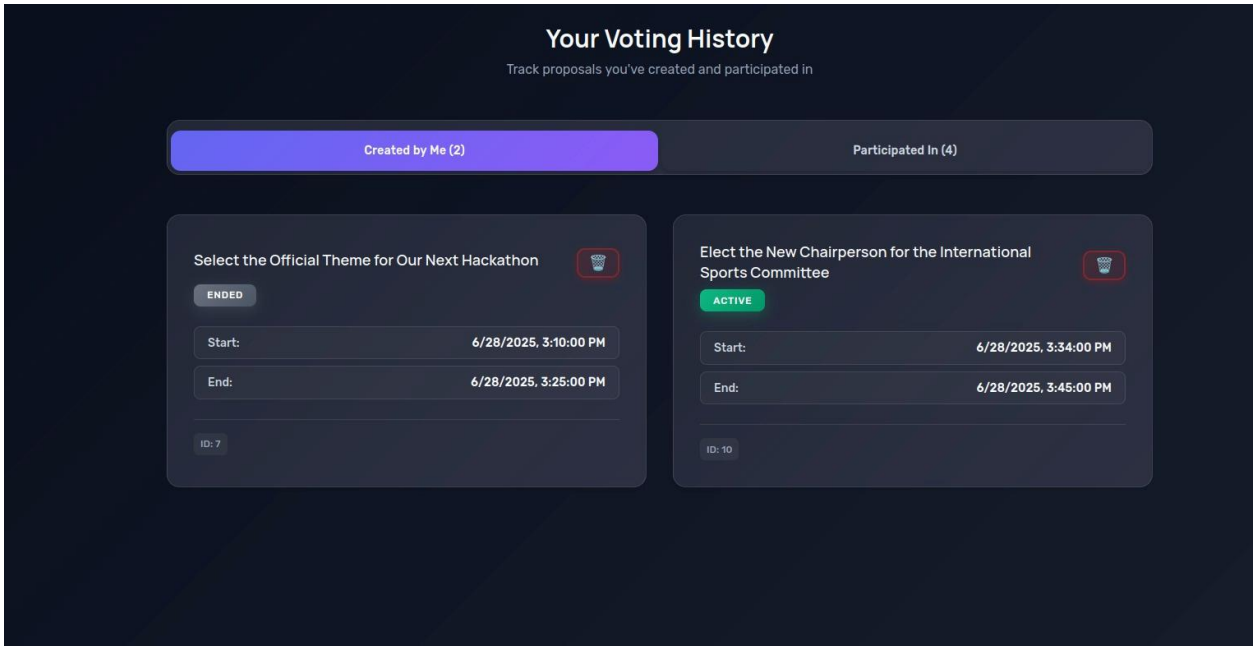
(Figure 5.7: Submit Immutable Vote)



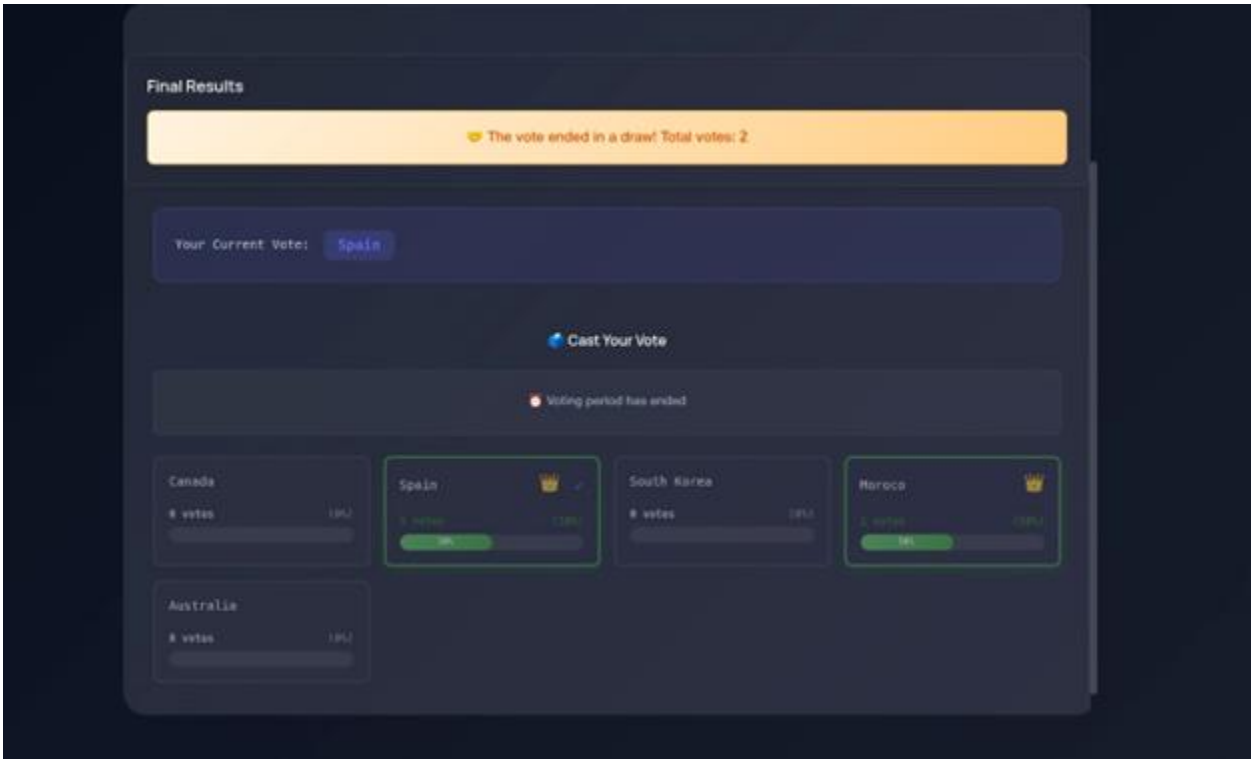
(Figure 5.8: Submit mutable Vote)



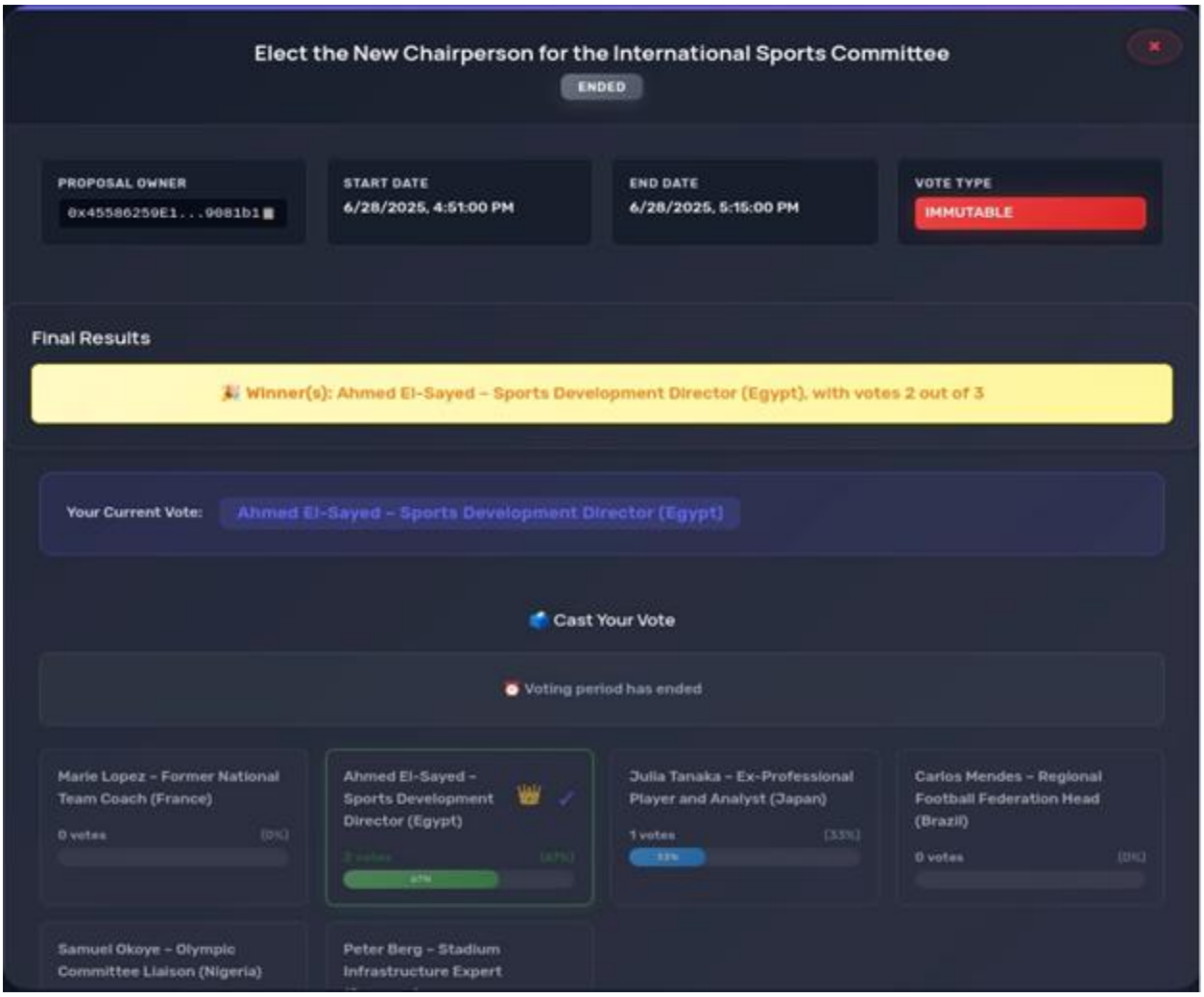
(Figure 5.8: Participation History)



(Figure 5.9: Proposal Creation History)



(Figure 5.10: Draw Case)



(Figure 5.11: Win Case)

Chapter Six:

Conclusion & Future Work

6.1 Conclusion

In this project, we designed and implemented a **secure, AI-powered blockchain-based e-voting system** tailored for verifying Egyptian citizens through their National ID cards. Our goal was to build a system that not only enhances voting **security** and **accuracy** but also **automates** the entire process, from user authentication to vote recording, while ensuring **accessibility** and **trust** for all participants.

Secure and Verified Identity Recognition

To ensure each vote came from a verified user, we incorporated a multi-layered identity verification pipeline:

- **Face Verification**: We employed deep learning–based face recognition techniques to compare live images against pre-registered ones, ensuring that only the true account holder could access the voting system. This was the first step to prevent identity fraud and impersonation.
- **Fake ID Detection**: Our system also detects and rejects fake or duplicate IDs through multiple AI checks.
- **National ID (NID) Extraction Using OCR**: A major innovation in our work is the integration of object detection (YOLO) with **EasyOCR** to extract the NID number directly from Egyptian national ID cards. This ensures consistency and accuracy when matching voter data with pre-registered government records.
    - We applied image preprocessing techniques such as grayscale conversion, inverse thresholding, resizing, and noise reduction to boost OCR accuracy.

- Post-processing was also implemented to convert Arabic digits to English and clean the extracted NID for accurate validation.

This layered verification system guarantees the principle of **"one account, one vote"**, securing the voting process from manipulation or unauthorized access.

AI-Powered Field Detection

Our system uses a two-stage YOLO detection pipeline:

1. **YOLO Model 1** detects the ID card from complex background images.
2. **YOLO Model 2** extracts specific fields such as name, NID, gender, and birthdate from the ID itself.

This allowed precise segmentation and extraction of only the necessary information, ensuring fast processing and reducing OCR errors.

Blockchain Integration with zkSync

To securely store and verify votes, we integrated our system with the **zkSync blockchain**, chosen for its **scalability**, **low gas fees**, and **zero-knowledge proof capabilities**. Every vote is:

- **Cryptographically recorded** as an immutable transaction.
- **Validated by smart contracts** to ensure voter eligibility and prevent double voting.
- **Tamper-proof and auditable**, maintaining full transparency throughout the election process.

This makes our system not only secure but also ready for large-scale real-world elections.

Dynamic Website for Usability

To ensure accessibility and ease of use, we developed a **dynamic and user-friendly website** that abstracts away blockchain complexity. This platform allows users to:

- Register and authenticate using face ID and NID.
- Submit their vote through a simple, guided interface.
- View confirmation and status securely without needing blockchain knowledge.

Our goal was to make **advanced security feel simple** for every user, ensuring wide adoption regardless of technical background.

Summary of Achievements

This project brought together several advanced technologies in a unified and innovative system:

- **First-time integration** of facial recognition, field-level OCR, and blockchain in a single secure voting platform.
- **Custom AI models** (YOLO-based) for ID field detection, integrated with EasyOCR and preprocessing to extract Arabic NID numbers with high accuracy.
- A **zkSync-based smart contract system** for scalable, verifiable vote storage instead of common Ethereum Blockchain.
- A **fully functional dynamic website** for usability and accessibility.

Together, these components created a full-stack secure e-voting system that is fast, scalable, fraud-resistant, and easy to use.

6.2 Future Work

While our system demonstrates a strong foundation for secure and automated blockchain-based e-voting, there are several areas for enhancement and future exploration:

- **OCR Error Handling and User Feedback Loop**
  Although our OCR module performs well with clear ID images, there are cases where text recognition may fail due to poor lighting, blur, or image quality. In future versions, if the OCR fails to extract the NID, the system will redirect the user to a **fallback page** where they are prompted to **upload a new image of their ID**. This improves reliability without disrupting the user experience.
- **End-to-End Liveness Detection**
  Adding advanced liveness detection (e.g., blink detection, head movement) during face verification can further protect against spoofing attacks such as photos or videos being used for login.
- **Offline Voting with Sync Support**
  Future versions could explore secure offline voting, where users can cast their vote without internet access and the system synchronizes securely once connectivity is restored.
- **Multi-modal Biometric Verification**
  Incorporating additional biometric factors (e.g., fingerprint or voice recognition) could further increase security and adapt to user preferences.
- **Integration with National Databases**
  To ensure real-time verification, future systems could be integrated directly

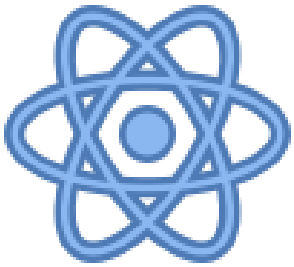
with government or civil registration databases (e.g., national ID registries), allowing direct cross-checks of user-submitted data.

- **Broader Applicability**

  Beyond voting, this architecture can be adapted to other civic services such as digital passport control, subsidy distribution, or secure digital identity issuance, using the same AI + blockchain foundation.

Our current system is a solid step forward, but with these enhancements, it could evolve into a fully scalable, fault-tolerant platform for secure digital governance.

Tools



References

[1] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, A. Zisserman, “VGGFace2: A dataset for recognising faces across pose and age”, *IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, IEEE, May 2018, pp. 67-74.

[2] Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, “Siamese Neural Networks for One-shot Image Recognition”, *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, JMLR, July 2015, Vol. 37, pp. 1-8.

[3] Florian Schroff, Dmitry Kalenichenko, James Philbin, "FaceNet: A Unified Embedding for Face Recognition and Clustering", *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, June 2015, pp. 815-823.

[4] Yi Sun, Xiaogang Wang, Xiaoou Tang, "Learning Face Representation from Scratch", *arXiv preprint arXiv:1411.7923*, November 2014.

[5] Xiaolong Wang, Abhinav Gupta, "Learning Robust Deep Face Representation", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, June 2015, pp. 5528-5537.

[6] Gaddam Harsha Vardhan, Swapnil Shah, Vanshika Gupta, Rohithreddy B. C., Tanya Bisht, "Voting System Using Blockchain (Face Recognition)", *International Journal of Engineering Research &*

Technology (IJERT), NCET - 2022 Conference Proceedings, June 2022, Volume 11, Issue 06, pp. 1-6.

[7] V. Sathya Preiya, V. D. Ambeth Kumar, R. Vijay, Vijay K., N. Kirubakaran, “Blockchain-Based E-Voting System with Face Recognition”, *International Journal of Intelligent Systems and Applications in Engineering*, ISSN: 2147-6799, April 2024, Vol. 12, No. 15s, pp. 240–250.

[8] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, Alex Alemi, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning", *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*, AAAI Press, February 2017, pp. 4278-4284.

[9] Irvine Mutandiwa, Calvin P. Mugauri, Maronge Musara, "E-Voting using Blockchain: Moving Away from the Ballot Paper", *International Journal of Scientific and Research Publications (IJSRP)*, Volume 12, Issue 5, May 2022, pp. 1-9, ISSN: 2250-3153.

[10] Bruno Miguel Batista Pereira, José Manuel Torres, Pedro Miguel Sobral, Rui Silva Moreira, Christophe Pinto de Almeida Soares, Ivo Pereira, "Blockchain-Based Electronic Voting: A Secure and Transparent Solution", *Journal of Information Systems Engineering & Management*, Volume 8, Issue 2, 2023, Article No. 22184, ISSN: 2468-4376.

[11] Achilleas Spanos, Ioanna Kantzavelou, "A Blockchain-based Electronic Voting System: EtherVote", *2020 IEEE International Conference on Blockchain (Blockchain)*, IEEE, November 2020, pp. 472-479.

[12] Arnab Mukherjee, Souvik Majumdar, Anup Kumar Kolya, Saborni Nandi, "A Privacy-Preserving Blockchain-based E-voting System", *Journal of Information Security and Applications*, Elsevier, Volume 70, March 2022, 103362, ISSN: 2214-2126.

[13] Zhengrui Huang, "CN-LBP: Complex Networks-based Local Binary Patterns for Texture Classification," IEEE/CAA Journal of Automatica Sinica, 2023.

[14] Sunen Chakraborty, Kingshuk Chatterjee, Paramita Dey, "Detection of Image Tampering Using Deep Learning, Error Levels and Noise Residuals," Neural Processing Letters, vol. 56, no. 112, pp. 1–16, March 2024.

[15] Rahmat Kurniawan, "Verification of ID Card using Optical Character Recognition (OCR): Case Study on Eligibility of Subsidy Recipients at PT PLN (Persero)", *Thesis*, Institut Teknologi Bandung, 2024.