Bilkent University

Department of Computer Engineering

Senior Design Project

T2424

IAWIA

Final Report

Emrehan Ateş - 22003465 - emrehan.ates@ug.bilkent.edu.tr

Eren Karakaş - 22002722 - eren.karakas@ug.bilkent.edu.tr

Fırat Utku Gül - 22003105 - utku.gul@ug.bilkent.edu.tr

Mehmet Emre Güneş - 22003287 - emre.gunes@ug.bilkent.edu.tr

Serhat Merak - 22002414 - serhat.merak@ug.bilkent.edu.tr

Supervisor: Uğur Güdükbay

Instructors: Atakan Erdem and Mert Bıçakçı

2 May 2025

Table of Contents

1. Introduction

1.1 Purpose of the System

In the digital age, secure online identity verification has become critical for services such as banking, healthcare, and social media. Traditional identity verification methods often store and share sensitive user data through centralized systems, leading to frequent data breaches and eroding trust between users and service providers [1]. The IAWIA (I Am Who I Am)

system addresses these issues by providing a decentralized, secure, and privacy-focused identity verification solution [2]. Utilizing advanced cryptographic techniques, such as zero-knowledge proofs (ZKP) and decentralized identity management, IAWIA enables users to prove their identities without disclosing unnecessary personal information [3]. This approach significantly reduces the risks of data leaks, unauthorized access, and centralized points of failure.

1.2 Design Goals

The IAWIA system is designed to be secure, efficient, and scalable. The main priorities are security and privacy, which are protected through various cryptographic techniques. The system is decentralized, which means there is no single point of failure, enhancing its security and reliability. Usability is another important goal; IAWIA aims to be simple and easy to use, creating a bridge between the everyday user and complex cryptographic solutions. Performance is considered to ensure that identity verification is fast. Reliability is another key factor; the system is built to work under different conditions without unexpected failures. Marketability is also taken into account, ensuring the system is helpful for businesses and follows industry trends. Extensibility allows future updates and improvements as new requirements emerge. Maintainability and flexibility help keep the system up to date with minimal effort.

1.3 Definitions, Acronyms, and Abbreviations

Know Your Customer (KYC) is a process used by businesses to confirm the identity of their customers.
A decentralized Identifier (DID) is a special kind of identifier that enables entities to be verifiably identified without needing a centralized database.
Public Key Infrastructure (PKI) is a security framework that helps with safe communication and authentication using cryptographic keys. It binds public keys with the identities of respective entities.
Zero-knowledge proof (ZKP) is a cryptographic method where one party can prove they know something without revealing the actual information [4].

1.4 Overview

This report presents the final version of the IAWIA system, encompassing all stages from refined analysis to design, implementation, testing, and maintenance planning. **Section 2** outlines the refined system requirements. **Section 3** details the final system architecture and design, including subsystem decomposition and security features. **Section 4** describes the implementation process, tools, technologies, and methodologies. **Section 5** presents the test cases that were conducted and the corresponding results. **Section 6** discusses the maintenance plan and future maintainability considerations. **Section 7** addresses broader engineering design factors, ethical and professional responsibilities, teamwork dynamics, and newly acquired knowledge applied during the project. **Section 8** concludes the report and outlines

potential directions for future work. **Sections 9 and 10** provide a glossary and a reference list to support the technical material discussed.

Through this structure, the report provides a comprehensive and self-contained overview of the IAWIA system, documenting its design, development, and realization.

2. Requirement Details

This section explains the main system requirements for the IAWIA project, including both functional and non-functional parts. It also mentions the changes we made during the development phase.

2.1 Functional Requirements

The IAWIA system needed to achieve these main functional requirements:

Passport Scanning: Users must be able to scan their passport using NFC technology on their mobile device. The system extracts necessary identity information securely.

Zero-Knowledge Proof (ZKP) Generation: The mobile app must generate a cryptographic proof (ZKP) that proves specific identity attributes (such as age or nationality) without showing sensitive details.

Identity Proof Management: The mobile app must store the identity proofs safely and allow users to manage their stored proofs.

Browser Extension Authentication: The browser extension must allow users to verify their identity on websites by sending a cryptographic proof instead of personal data.

Off-Chain Verification: The application will use off-chain verification to eliminate redundant costs and avoid performance overhead.

Secure Communication: Communication between the web client and browser extension must be encrypted to protect identity data during the transfer.

IPFS Storage: ZKP commitments will be stored on IPFS to ensure decentralized, immutable, and reusable access for efficient proof validation and reduced recomputation.

All of these functional requirements were fully implemented. We moved the ZKP generation off-chain to make it more performant and easier to use.

2.2 Non-Functional Requirements

The IAWIA system also aimed to meet important non-functional requirements:

Security: Security was the top priority. The system had to protect user identity with strong encryption, secure storage, and blockchain verification. We fully achieved this goal.

Privacy: User privacy was critical. The system had to ensure that only the absolutely necessary information was revealed during identity verification and that no personal information was stored in centralized servers. This goal was fully achieved.

Usability: The system should be easy to use, especially for people who are not experts in technology. We focused on simple app interfaces and easy authentication flow. We met this goal successfully.

Performance: Identity verification needed to be fast, ideally taking only a few seconds. We achieved this goal by moving some operations from the blockchain to traditional off-chain computation.

Scalability: The system should work well even if many users verify their identity simultaneously. Our decentralized design helped with this, although large-scale testing was beyond the scope.

Reliability: The system should work correctly under different conditions without unexpected crashes. During testing, the system showed good reliability.

Extensibility and Maintainability: The design needed to allow future updates, like supporting new types of ID documents or using different blockchain networks. Our modular design makes future improvements possible.

2.3 Changes from the Original Plan

During development, two main changes happened:

On-Chain to Off-Chain: The original verification process happened on the blockchain. However, we have decided to run ZKPs off-chain due to performance and economic constraints.

Wallet Integration: Instead of requiring a mobile device for functionality, we have integrated a browser wallet extension to streamline desktop usage.
Besides these two simplifications, the rest of the system followed the original plan closely, and all main services were completed.

3. Final Architecture and Design Details

This section provides the finalized version of the IAWIA system architecture and design. It includes the final architecture and system decomposition. We have changed some crucial points in the high-level flow during the last two semesters and finalized the architecture. Our design has two main flows, which are the enrollment phase and the verification phase.

3.1 Enrollment Phase

Enrollment is the process by which a new user (or an existing user's new identity attributes) is introduced into the IAWIA system. The goal of enrollment is to create a persistent commitment to the user's identity data without revealing the data itself and to anchor this commitment on the blockchain for future reference. Enrollment involves the following steps:

1. MRZ scan. The mobile application optically scans the machine-readable zone (MRZ) to derive Basic-Access-Control (BAC) keys that unlock the passport chip.
2. Chip read (NFC). Using the BAC session, the phone reads mandatory data groups (DG 1 & DG 2) and the Security Object Document (SOD). The SOD contains SHA-512 hashes of each DG and is signed by the digital signing certificate (DSC) provided by Country Signing Certification Authority (CSCA).
3. Blob transfer. The mobile application transmits the raw passport blob —including the DSC and SOD— to the IAWIA backend API over TLS. No data is persistently stored server-side.
4. Zero-knowledge integrity proof. The API runs a Groth16 circuit that proves, with zero knowledge, that (i) every DG hash matches the SOD, (ii) the SOD is correctly signed by the DSC, and (iii) the DSC chains to a trusted CSCA public key. A single "integrity" bit is returned to the device.
5. Commitment generation. Upon success, the app samples a 256-bit secret s and computes a binding commitment $C = \text{SHA-512}(DG1 \;\|\; DG2 \;\|\; s)$. The secret s is sealed in the device's hardware keystore.
6. Commitment persistence. The commitment file —containing C, the circuit verification key, and non-identifying metadata— is uploaded to IPFS.
7. CID receipt. IPFS returns the multihash content identifier (CID) that uniquely addresses the file.
8. Ethereum key derivation. The app derives an Ethereum externally-owned account and generates a fresh key pair.
9. Key hand-off. The public key and an encrypted private key backup are transferred to the browser extension via a secure channel. In a later step (outside the enrollment phase), the extension publishes the CID in a minimal ETH-burn transaction, thereby time-stamping the commitment on-chain.

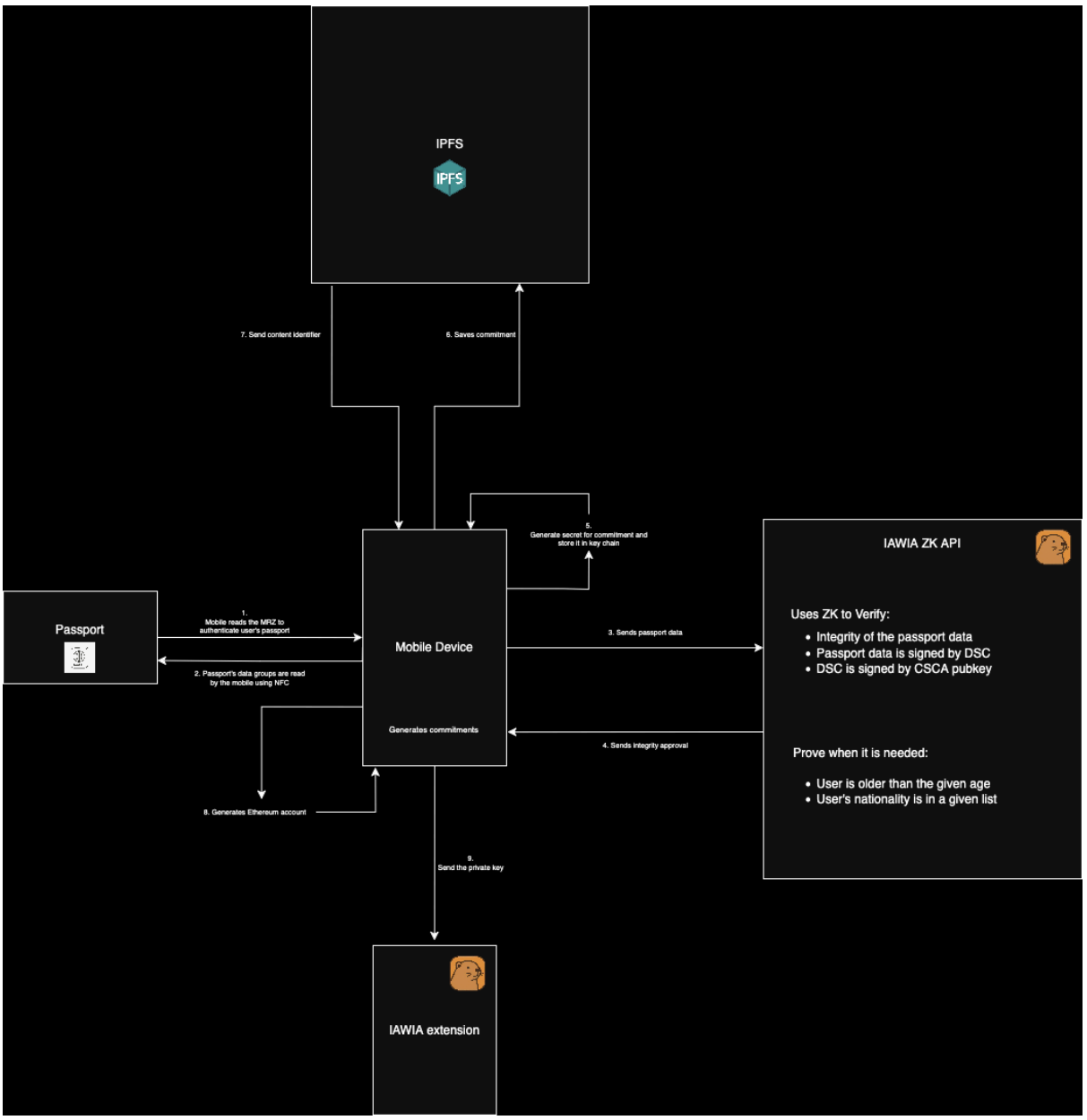


Figure 1: Enrollment Flow

3.2 Verification Phase

The verification phase enables a previously enrolled user to prove possession of a committed identity credential to an external relying party without revealing any underlying private data.

As illustrated in Figure 2, the architecture comprises four principal components:

The Third-Party Backend, which requests proof of identity or attributes.

The Web Client, which mediates between the relying party and the user's browser extension.

The IAWIA Browser Extension, which manages credential retrieval and proof orchestration.

The IAWIA Backend, which executes the computationally intensive zero-knowledge proof (ZKP) generation and optional verification.

The protocol unfolds in seven discrete steps:

1. Proof Request Retrieval. The Web Client issues an HTTP request to the Third-Party Backend, soliciting the proof parameters that define the disclosure scope (e.g., attribute predicates or credential types).
2. Parameter Relay. Upon receipt, the Web Client forwards these proof parameters to the IAWIA Browser Extension via a secured message channel.
3. Credential Identifier Extraction. The Extension invokes the user's local wallet to obtain the stored Content Identifier (CID) corresponding to the enrolled identity commitment.
4. This CID points to an IPFS file anchoring the user's public commitment and associated verification metadata.
5. Proof Generation Request. The Extension assembles the public inputs (including the disclosure parameters and any challenge nonce) alongside the CID and transmits them to the IAWIA Backend over TLS.
6. Off-Chain ZKP Construction. The IAWIA Backend retrieves the commitment file from IPFS using the CID, loads the appropriate pre-compiled ZK-SNARK circuit, and generates a proof attesting that the user's secret attributes satisfy the committed statement without revealing them. This step leverages optimized cryptographic libraries to ensure subsecond performance.
7. Proof Delivery. The computed proof—and, if chosen, the corresponding public signals—is returned to the Browser Extension. Optionally, the backend may verify the proof locally before delivery to simplify the extension's responsibilities.
8. Proof Forwarding and Final Verification. Finally, the Extension transmits the proof to the Web Client, which relays it back to the Third-Party Backend. The relying party performs a final cryptographic check against the commitment metadata; upon success, the user is granted access to the protected resource.

By confining sensitive operations like retrieval of secret inputs and proof verification within the user's browser extension and delegating heavy-weight proof construction to a trusted off-chain service, the IAWIA architecture preserves user privacy, minimizes client-side overhead, and entirely avoids on-chain transactions during authentication—thus eliminating gas costs and blockchain latency from the verification workflow.

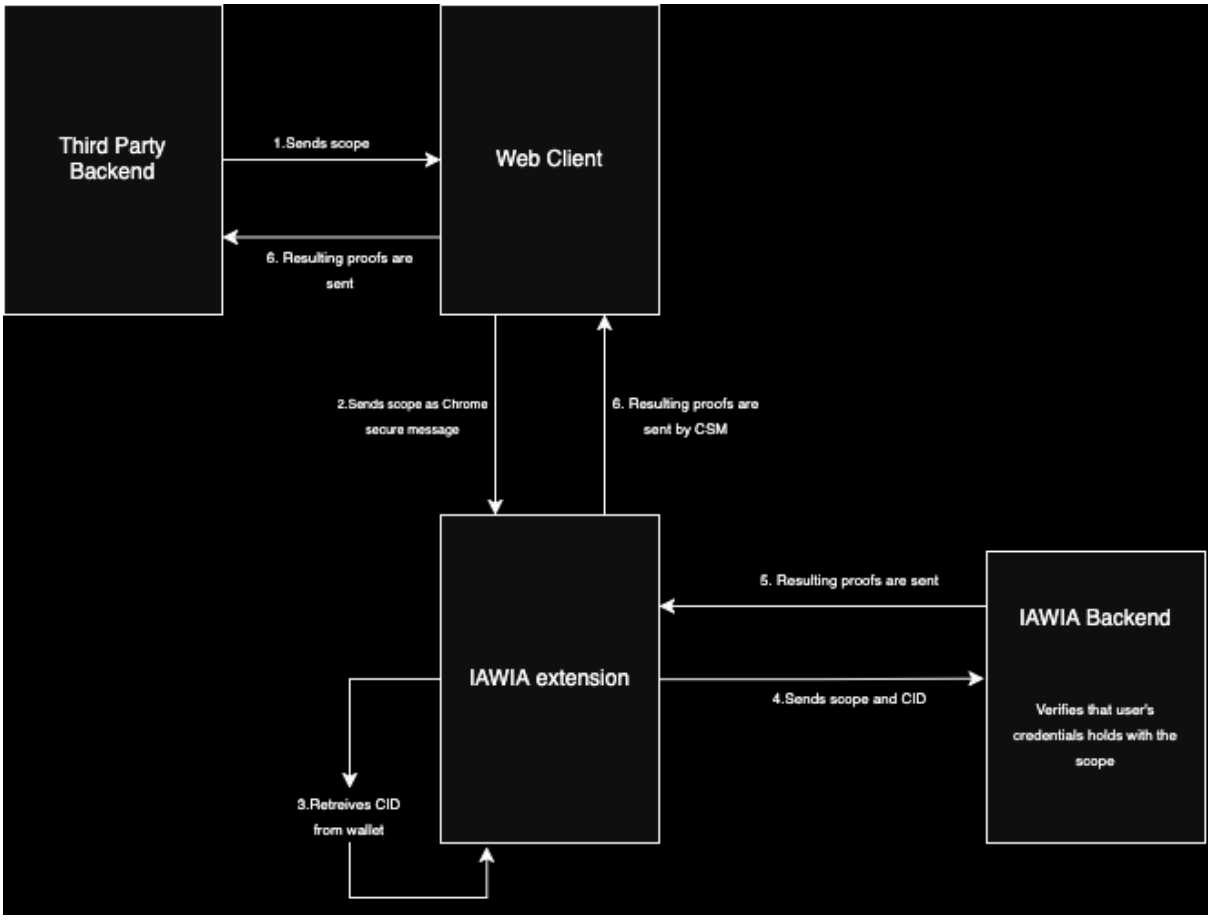


Figure 2: Verification Flow

3.3 Subsystem Decomposition

In the final implementation, the IAWIA system was organized into three major subsystems:

3.3.1 Mobile Application Subsystem

Passport Scanning: The app scans NFC-enabled passports to extract identity data.

Attribute Extraction: Key identity information such as name, date of birth, and nationality is extracted and used to generate cryptographic commitments.

Zero-Knowledge Proof Generation: The extracted data is used to generate cryptographic proofs that allow verification without revealing the raw data.

Proof Management: Users can securely store and manage their cryptographic commitments within the IPFS.

Local Storage: Sensitive data is stored locally on the device in an encrypted format, ensuring no data is transmitted or stored on external servers.

3.3.2 Browser Extension Subsystem

Verification Request Handling: When a website requests verification, the extension communicates with the blockchain to retrieve CID.

Proof Forwarding: The browser extension forwards the proof to the requesting site, enabling verification without revealing identity details.

User Consent:Users must manually approve all verification requests, ensuring they retain control and awareness over their data.

3.3.3 Blockchain Subsystem

CID Storage: Only CIDs of commitments are stored on the blockchain, ensuring that no personal data is exposed.

Decentralization: The verification process is decentralized, meaning no central authority has control over the validation process [10].

3.5 Final Observations and Conclusion

After the full implementation of IAWIA, we can confirm that the system succeeded in achieving its original goals:

- **Fully Decentralized**: No central database exists. Users control their data.
- **Strong Privacy**: Zero-knowledge proofs protect sensitive attributes.
- **High Security**: Local encryption, blockchain verification, secure communication channels.
- **User Friendly**: The mobile app and browser extension provide an easy-to-use experience.

Some minor technical improvements, like stronger communication encryption before proof usage, were made during the implementation. However, the general architecture and design were successfully followed almost exactly as planned.

With IAWIA, users can now verify their identities online without having to give away their real personal information, which is a big success for digital privacy and decentralized identity systems.

4. Development/Implementation Details

During the development phase of the IAWIA project, each part of the system was built carefully using a variety of modern technologies, frameworks, and programming languages. Our architecture consists mainly of three major parts: the mobile application, the browser extension, and the backend server. Each part required different tools and approaches to achieve a secure, decentralized, and user-friendly identity verification system.

The following sections will explain how each major part was developed, mentioning the critical technologies, libraries, and design decisions made throughout the project.

4.1 Mobile Application

The mobile application plays a very important role in the IAWIA system. It is responsible for scanning user passports through NFC, generating commitments, and managing user identity attributes locally.

For the development of the mobile app, the main technologies used were:

TypeScript: The main logic of the mobile application, especially parts dealing with app functionality, form validations, and basic user interfaces, was written using TypeScript. TypeScript was chosen because of its type system, which helped reduce bugs and made the code more reliable and easier to maintain.

Kotlin: Kotlin was used for some platform-specific operations, especially when accessing Android-specific features such as NFC hardware. Since NFC interactions are critical for passport scanning, implementing these parts natively in Kotlin ensured better performance and reliability on Android devices.

Other: Small scripts were written in Shell or Python to automate build processes.

Important Development Decisions for Mobile App

Local Secure Storage: After commitments are generated, secrets are stored encrypted inside the device using secure storage solutions. This decision protects user data even if the device was compromised.

NFC Passport Scanning: For the NFC scanning of passports, the app had to request user permissions properly and use cryptographic checks to ensure that the passport chip data was genuine and not tampered with.

Overall, the mobile application development required strong integration between cryptography, secure hardware access, and user interface design to create a smooth and safe experience for users.

4.2 Browser Extension

The second major part of the system is the browser extension, which acts as the user's wallet for managing identity proofs and sharing them selectively with websites that request identity verification.

The browser extension was developed mainly using the following:

TypeScript: TypeScript was the dominant language for the extension. It allowed us to write secure, scalable, and easy-to-read code for the extension, popup interfaces, and communication layers.

CSS: CSS was used to style the extension's user interface. Our goal was to keep the design simple, clean, and intuitive so users could easily view, select, and share their identity proofs.

HTML: HTML was used to structure the extension's popup UI and background pages.

Important Development Decisions for Browser Extension

Secure Communication: We implemented secure messaging between the browser extension and the web client. Chrome Secure Messaging APIs ensured end-to-end encryption when proofs were requested or shared.

User Consent: Before sending any proof to a website, the browser extension always asks for explicit user consent. This ensures that no proof can be leaked without the user's knowledge or approval.

Proof Request Handling: When a service requests identity verification, the extension fetches the necessary proof from the mobile app rather than storing it inside the extension. This decision minimizes the amount of sensitive data stored in the browser environment.

The browser extension became a powerful but safe tool for decentralized identity management by focusing on privacy and security at every step.

4.3 Backend Server

The third major component is the backend API, where arithmetic circuits are stored and run.

The backend was developed using:

TypeScript: Routers and endpoints are implemented using TypeScript. Bash scripts for running the circuits are automated through TypeScript's interface.

Circom: Circom is a domain-specific language (DSL) for writing arithmetic circuits, which are necessary for generating zero-knowledge proofs. We used Circom extensively to define the logic for verifying identity attributes without revealing private information. The Circom circuits were compiled and integrated into the app, enabling proof generation on the user's device.

Important Development Decisions for Backend Server

Zero-Knowledge Proof Circuits: One major decision was to perform all ZKP generation off-chain rather than on the blockchain. Although this reduces privacy guarantees, making the application performance viable is important, as proof generation is computationally expensive.

Minimal On-Chain Data: To protect user privacy and reduce gas costs, we decided not to store any real user data on the blockchain. Instead, only cryptographic hashes were stored, which are enough to associate identity proofs with users without revealing sensitive information.

5. Test Cases and Results

5.1 Passport Scanning Service

Test ID	001	**Category**	Functional	**Severity**	Critical
Objective	Ensure that the system can successfully detect and scan a passport via NFC.				
Steps	1. Launch the IAWIA mobile application. 2. Navigate to the passport scanning feature. 3. Enable NFC on the mobile device. 4. Place an NFC-enabled passport near the phone's NFC reader. 5. Observe whether the system detects the chip and initiates scanning.				
Expected	The system successfully detects and initiates the passport scanning process.				
Date-Result	01/05/2025 - Passed				

Test ID	002	**Category**	Functional	**Severity**	Critical
Objective	Verify that the system correctly extracts the necessary identity attributes from a valid passport.				
Steps	1. Launch the IAWIA mobile application. 2. Navigate to the passport scanning feature. 3. Enable NFC on the mobile device. 4. Place a valid NFC-enabled passport near the phone's NFC reader. 5. Wait for the system to process the passport data.				
Expected	The system successfully extracts the passport holder's name, date of birth, nationality, and passport number. No incorrect data is received.				
Date-Result	01/05/2025 - Passed				

Test ID	003	**Category**	Functional	**Severity**	High
Objective	Ensure that the system appropriately handles scanning failures caused by invalid or damaged passports.				
Steps	1. Launch the IAWIA mobile application. 2. Navigate to the passport scanning feature. 3. Enable NFC on the mobile device. 4. Place an invalid or physically damaged passport near the phone's NFC reader. 5. Observe the system's response.				
Expected	The system detects that the passport is invalid or unreadable. A user-friendly error message is displayed, prompting the user to retry or use a different passport.				
Date-Result	01/05/2025 - Passed				

Test ID	004	**Category**	Security	**Severity**	Critical
Objective	Verify that the system does not store any raw passport data after generating a zero-knowledge proof.				
Steps	1. Scan a valid passport using the IAWIA mobile application. 2. Allow the system to generate a zero-knowledge proof. 3. Attempt to retrieve raw passport data from local storage.				
Expected	The system securely deletes the extracted passport data after generating the proof. No raw passport data remains stored on the device.				
Date-Result	01/05/2025 - Passed				

Test ID	005	**Category**	Functional	**Severity**	High
Objective	Ensure that the system properly handles failures related to encrypted passport chips.				
Steps	1. Launch the IAWIA mobile application. 2. Navigate to the passport scanning feature. 3. Place a passport with an encrypted chip near the NFC reader. 4. Observe the system's response.				
Expected	If the system cannot decrypt the passport data, it displays an appropriate error message. The application does not proceed with identity proof generation in case of decryption failure.				
Date-Result	01/05/2025 - Passed				

Test ID	006	**Category**	Security	**Severity**	Critical
Objective	Ensure that the system verifies the authenticity of the passport before extracting data.				
Steps	1. Launch the IAWIA mobile application. 2. Navigate to the passport scanning feature. 3. Scan a tampered or counterfeit passport. 4. Observe whether the system detects anomalies.				
Expected	The system verifies the passport's authenticity before extracting data. If the passport is counterfeit, the system rejects it and displays an appropriate warning.				
Date-Result	01/05/2025 - Passed				

5.2 Zero-Knowledge Proof (ZKP) Generation Service

Test ID	007	**Category**	Functional	**Severity**	Critical
Objective	Ensure that the system successfully generates zero-knowledge proofs (ZKPs) from valid identity attributes.				
Steps	1. Launch the IAWIA mobile application. 2. Navigate to the ZKP generation feature. 3. Scan a valid passport to extract identity attributes. 4. Initiate ZKP generation using the extracted data.				

Expected	The system successfully generates a ZKP based on the extracted identity attributes. The ZKP is cryptographically valid and usable.
Date-Result	01/05/2025 - Passed

Test ID	008	**Category**	Functional	**Severity**	Critical
Objective	Ensure that generated ZKPs are correct and valid for verification.				
Steps	1. Generate a ZKP using a valid identity attribute set. 2. Use the system's verification function to validate the ZKP. 3. Compare the verification result with expected correctness criteria.				
Expected	The ZKP passes the verification check. The verification process does not reveal raw identity attributes.				
Date-Result	01/05/2025 - Passed				

Test ID	009	**Category**	Functional	**Severity**	High
Objective	Ensure that users can selectively disclose identity attributes through ZKPs.				
Steps	1. Launch the IAWIA mobile application. 2. Navigate to the selective disclosure feature. 3. Choose specific identity attributes to include in the ZKP. 4. Generate a ZKP with the selected attributes.				
Expected	The system generates a ZKP containing only the selected attributes. Unselected attributes are not used in proof generation.				
Date-Result	01/05/2025 - Passed				

Test ID	010	**Category**	Security	**Severity**	Critical
Objective	Ensure that the system detects and rejects tampered or invalid identity attributes.				
Steps	1. Modify identity attributes before ZKP generation such as changing the name or the passport number. 2. Attempt to generate a ZKP with the altered data. 3. Verify whether the system detects tampering.				
Expected	The system identifies the tampered identity attributes. ZKP generation is rejected with an appropriate error message.				

Date-Result	01/05/2025 - Passed

Test ID	011	**Category**	Performance	**Severity**	High
Objective	Ensure that ZKP generation completes within an acceptable time on mobile devices.				
Steps	1. Launch the IAWIA mobile application. 2. Initiate ZKP generation using valid identity attributes. 3. Measure the time taken to generate the ZKP.				
Expected	The ZKP generation process completes within a predefined time threshold. The application remains responsive during ZKP computation.				
Date-Result	01/05/2025 - Passed				

Test ID	012	**Category**	Security	**Severity**	Critical
Objective	Ensure that generated ZKPs are securely stored and protected from unauthorized access.				
Steps	1. Generate a ZKP using a valid identity attribute set. 2. Attempt to access stored ZKPs using unauthorized methods. 3. Observe whether the system prevents unauthorized access.				
Expected	ZKPs are securely stored using encryption or other security measures. Unauthorized access attempts are blocked.				
Date-Result	01/05/2025 - Passed				

Test ID	013	**Category**	Security	**Severity**	High
Objective	Ensure that expired or revoked ZKPs are not valid for authentication.				
Steps	1. Generate a ZKP with a predefined expiration time. 2. Attempt to use the ZKP after its expiration time. 3. Observe whether the system correctly invalidates the ZKP.				
Expected	Expired ZKPs cannot be used for authentication. Revoked ZKPs are immediately invalidated.				
Date-Result	01/05/2025 - Passed				

5.3. Identity Proof Management Service

Test ID	014	**Category**	Functional	**Severity**	High
Objective	Ensure users can view their stored identity proofs within the mobile application.				
Steps	1. Launch the IAWIA mobile application. 2. Navigate to the "Identity Proofs" section. 3. Select a stored identity proof to view its details.				
Expected	The list of stored identity proofs is displayed correctly. The selected identity proof's details are visible without revealing sensitive information unnecessarily.				
Date-Result	01/05/2025 - Passed				

Test ID	015	**Category**	Functional	**Severity**	High
Objective	Ensure users can add new identity attributes and update existing proofs.				
Steps	1. Open the "Identity Proofs" section in the mobile app. 2. Select an existing identity proof 3. Add a new identity attribute. 4. Save the updated proof.				
Expected	The new identity attribute is successfully added to the identity proof. The updated proof is stored securely.				
Date-Result	01/05/2025 - Passed				

Test ID	016	**Category**	Security	**Severity**	Critical
Objective	Ensure identity proofs are deleted permanently and securely.				
Steps	1. Navigate to the "Identity Proofs" section in the mobile app. 2. Select an identity proof to delete. 3. Confirm the deletion. 4. Attempt to access the deleted proof.				
Expected	The selected identity proof is permanently removed. Deleted data cannot be recovered through regular app usage. Secure deletion mechanisms are applied.				
Date-Result	01/05/2025 - Passed				

Test ID	017	**Category**	Security	**Severity**	Critical
Objective	Ensure cryptographic commitments are stored encrypted in the IPFS.				
Steps	1. Generate a zero-knowledge proof. 2. Attempt to access the commitment data directly in IPFS. 3. Verify whether the data is encrypted.				
Expected	Commitment data is encrypted at rest. Unauthorized access to raw data is prevented.				
Date-Result	01/05/2025 - Passed				

Test ID	018	**Category**	Functional	**Severity**	Medium
Objective	Ensure users can restore their identity proofs after reinstalling the app.				
Steps	1. Store identity proofs in the mobile application. 2. Uninstall and reinstall the application. 3. Log in to the app with the same account. 4. Check if stored identity proofs are restored.				
Expected	Identity proofs are restored on reinstallation.				
Date-Result	01/05/2025 - Passed				

5.4 Browser Extension Authentication Service

Test ID	019	**Category**	Functional	**Severity**	High
Objective	Ensure that the browser extension can successfully request authorization from the web client.				
Steps	1. Install and activate the browser extension. 2. Navigate to a website that requires identity verification. 3. Click on the appropriate component and initiate a verification request. 4. Observe if the request is sent to the extension.				
Expected	The extension receives a notification concerning the authorization request. The request includes correct identity attributes as required by the website.				
Date-Result	01/05/2025 - Passed				

Test ID	020	**Category**	Functional	**Severity**	High
Objective	Ensure users can approve identity verification requests from the extension.				
Steps	1. Receive an identity verification request on the extension. 2. View the request details, including the website requesting verification. 3. Select “Approve”. 4. Observe the browser extension’s behavior based on the user’s response.				
Expected	Approving the request successfully authenticates the user on the website.				
Date-Result	01/05/2025 - Passed				

Test ID	021	**Category**	Functional	**Severity**	High
Objective	Ensure users can reject identity verification requests from the extension.				
Steps	1. Receive an identity verification request on the extension. 2. View the request details, including the website requesting verification. 3. Select “Reject”. 4. Observe the browser extension’s behavior based on the user’s response.				
Expected	The extension denies authentication and notifies the requesting website. No authentication occurs if the request is ignored.				
Date-Result	01/05/2025 - Passed				

Test ID	022	**Category**	Security	**Severity**	Critical
Objective	Verify that the browser extension does not store identity proofs locally.				
Steps	1. Complete multiple identity verifications using the browser extension. 2. Inspect browser storage for identity proof data. 3. Restart the browser and attempt to retrieve previously verified identity proofs.				
Expected	The browser extension does not retain identity proof data after authentication. No personally identifiable information is found in browser storage.				
Date-Result	01/05/2025 - Passed				

Test ID	023	**Category**	Security	**Severity**	Critical
Objective	Ensure that unauthorized identity verification requests are blocked.				

Steps	1. Log out of the mobile application. 2. Attempt to request identity verification from the browser extension. 3. Try to send a forged identity verification request without a linked mobile application.
Expected	The browser extension does not send identity verification requests without an active mobile session. The mobile app rejects verification requests from unrecognized or unauthorized extensions.
Date-Result	01/05/2025 - Passed

Test ID	024	**Category**	Security	**Severity**	Critical
Objective	Ensure that the browser extension can detect and block fake verification requests from phishing websites.				
Steps	1. Attempt to use the browser extension on a legitimate website. 2. Identify a phishing website that mimics the legitimate one. 3. Attempt identity verification on the phishing site.				
Expected	The extension detects suspicious or unverified websites and alerts the user. The mobile app warns users if a request comes from an untrusted source.				
Date-Result	01/05/2025 - Passed				

5.5 Secure Communication Service

Test ID	025	**Category**	Security	**Severity**	Critical
Objective	Ensure that an encrypted communication channel is properly established between the web client and browser extension.				
Steps	1. Open the web client and browser extension. 2. Initiate a secure connection between them. 3. Inspect the communication protocol used for data exchange. 4. Monitor network requests.				
Expected	A secure channel is successfully established using industry-standard encryption. Sensitive data is transmitted only after encryption is confirmed. No unencrypted data is transmitted over the network.				
Date-Result	01/05/2025 - Passed				

Test ID	026	**Category**	Security	**Severity**	High
Objective	Verify that WebSocket communication between the mobile app and browser extension is encrypted and secure.				
Steps	1. Initiate a WebSocket connection between the mobile app and browser extension. 2. Analyze network traffic for encryption standards. 3. Attempt to intercept the communication using network sniffing tools. 4. Verify that no unencrypted messages are exchanged.				
Expected	All WebSocket communication is encrypted end-to-end. Attempts to intercept communication result in unreadable encrypted data. No identity-related data is leaked in plaintext.				
Date-Result	01/05/2025 - Passed				

Test ID	027	**Category**	Reliability	**Severity**	High
Objective	Ensure the system can handle network failures without compromising authentication.				
Steps	1. Start an authentication request between the mobile app and browser extension. 2. Disconnect the network midway. 3. Reconnect the network after a delay. 4. Attempt to resume or restart the authentication process.				
Expected	The system detects the network failure and informs the user. Authentication resumes securely after reconnection. If a timeout occurs, the authentication request must be restarted.				
Date-Result	01/05/2025 - Passed				

Test ID	028	**Category**	Security	**Severity**	Critical
Objective	Ensure that cryptographic proofs remain unaltered during transmission.				
Steps	1. Generate a cryptographic proof in the mobile app. 2. Store the proof in IPFS. 3. Modify or corrupt the proof during transmission using network manipulation tools. 4. Verify if the system detects the tampered proof.				
Expected	The system verifies the proof's integrity using cryptographic hash checks. Altered proofs are rejected, and authentication fails safely. The user is alerted in case of proof corruption.				

Date-Result	01/05/2025 - Passed

5.6 Blockchain-Based Verification Service

Test ID	029	**Category**	Functionality and Security	**Severity**	Critical
Objective	Ensure that CIDs are securely and correctly stored on the blockchain.				
Steps	1. Generate a zero-knowledge proof (ZKP) for an identity attribute in the mobile application. 2. Submit the proof commitment to the IPFS. 3. Confirm the CID anchor transaction on the blockchain explorer. 4. Verify that the CID is immutably stored.				
Expected	The commitment CID is successfully stored on the blockchain. The blockchain transaction contains the expected CID. Unauthorized modifications or deletions are not possible.				
Date-Result	01/05/2025 - Passed				

Test ID	030	**Category**	Functionality	**Severity**	High
Objective	Ensure that smart contracts can correctly retrieve and validate CIDs.				
Steps	1. Submit a ZKP to the extension for verification. 2. Call the extension function responsible for ZKP verification. 3. Observe whether the CID correctly leads to the commitment. 4. Check the blockchain logs to confirm validation success or failure.				
Expected	The IPFS communication correctly retrieves the necessary commitment data. The verification process completes successfully for valid proofs. Invalid proofs are correctly rejected.				
Date-Result	01/05/2025 - Passed				

Test ID	031	**Category**	Reliability	**Severity**	High
Objective	Ensure that the system handles network congestion or blockchain failures gracefully.				
Steps	1. Submit a transaction during a period of blockchain congestion. 2. Observe transaction confirmation times and potential delays.				

	3. Simulate a blockchain failure by using an offline node or network disruption. 4. Attempt to retry the transaction or implement a fallback mechanism.
Expected	The system informs users about transaction delays. Transactions are retried or queued for later processing. The application does not crash or lose data due to blockchain failures.
Date-Result	01/05/2025 - Passed

Test ID	032	**Category**	Security	**Severity**	Critical
Objective	Ensure that identity proofs cannot be reused maliciously in replay attacks.				
Steps	1. Generate a valid ZKP and submit it for verification. 2. Capture the transaction details and replay it on the blockchain. 3. Observe whether the system accepts the replayed proof. 4. Implement nonce-based or timestamped verification to prevent reuse.				
Expected	Replayed proofs are detected and rejected. Each proof verification requires a fresh, one-time nonce. The system prevents duplicate transactions from being accepted.				
Date-Result	01/05/2025 - Passed				

Test ID	033	**Category**	Security and Functionality	**Severity**	High
Objective	Ensure that expired or revoked identity proofs are not valid for verification.				
Steps	1. Store an identity proof with an expiration timestamp. 2. Attempt to verify the proof after the expiration period. 3. Revoke an identity proof manually and attempt verification. 4. Observe the system's response to expired or revoked proofs.				
Expected	Expired proofs are rejected during verification. Revoked proofs cannot be used for authentication. Users are notified if their identity proof is no longer valid.				
Date-Result	01/05/2025 - Passed				

5.7 Access Control and Security

Test ID	034	**Category**	Security	**Severity**	Critical
Objective	Ensure that users must authenticate using PIN before accessing the application.				
Steps	1. Launch the mobile application. 2. Attempt to access identity proofs without authentication. 3. Authenticate using PIN. 4. Observe the application's behavior when authentication succeeds or fails.				
Expected	Users must authenticate before accessing the app. Failed authentication attempts block access. The app remains locked after repeated failed attempts.				
Date-Result	01/05/2025 - Passed				

Test ID	035	**Category**	Security	**Severity**	Critical
Objective	Ensure that brute-force attacks are mitigated through rate limiting and lockout policies.				
Steps	1. Enter incorrect PIN multiple times. 2. Observe whether the app enforces a temporary lockout or increasing delay. 3. Monitor if authentication attempts can be automated.				
Expected	Multiple failed attempts trigger a temporary lockout. Automated brute-force attempts are blocked. Biometric authentication locks out after several failed tries.				
Date-Result	01/05/2025 - Passed				

Test ID	036	**Category**	Security	**Severity**	High
Objective	Ensure that cryptographic key pairs are securely generated and stored.				
Steps	1. Generate a new cryptographic key pair in the app. 2. Attempt to extract private keys from local storage. 3. Verify that keys are stored in a secure enclave or hardware-backed keystore.				

	4. Test whether unauthorized apps can access the keys.
Expected	Private keys are securely generated and stored. Keys are inaccessible to unauthorized applications. Key material cannot be extracted from the device.
Date-Result	01/05/2025 - Passed

Test ID	037	**Category**	Security	**Severity**	High
Objective	Ensure that identity proofs are signed securely before verification.				
Steps	1. Generate a zero-knowledge proof (ZKP) for an identity attribute. 2. Sign the proof using the user's private key. 3. Submit the proof for verification. 4. Observe whether verification succeeds with a valid signature. 5. Modify the proof and attempt verification with the same signature.				
Expected	Proofs are signed securely before submission. Modified proofs fail verification due to signature mismatch. No unauthorized signing is allowed.				
Date-Result	01/05/2025 - Passed				

Test ID	038	**Category**	Security	**Severity**	Critical
Objective	Ensure that lost or compromised keys do not allow unauthorized access.				
Steps	1. Revoke the current key pair in case of compromise. 2. Generate a new key pair and re-enroll identity proofs. 3. Attempt to use the old key for signing and verification. 4. Verify that only the new key is accepted.				
Expected	Lost keys can be revoked and replaced. Old keys become invalid immediately. Users are notified of key compromise events.				
Date-Result	01/05/2025 - Passed				

Test ID	039	**Category**	Security	**Severity**	Critical
Objective	Ensure that sensitive identity data remains encrypted during storage and transmission.				
Steps	1. Capture network traffic during proof transmission. 2. Attempt to read identity attributes from intercepted data. 3. Inspect stored identity proofs in local storage.				

	4. Verify encryption of data at rest and in transit.
Expected	All sensitive data is encrypted end-to-end. Identity attributes are not exposed in plaintext. Data leaks are prevented during transmission and storage.
Date-Result	01/05/2025 - Passed

Test ID	040	**Category**	Security	**Severity**	Critical
Objective	Ensure that replayed verification requests are detected and blocked.				
Steps	1. Capture a valid identity verification request. 2. Replay the same request to the authentication system. 3. Observe whether the system accepts or rejects it. 4. Implement timestamped verification.				
Expected	Replayed verification requests are detected and rejected. Timestamp validation prevents duplicate authentication. Verification logs show evidence of replay attack prevention.				
Date-Result	01/05/2025 - Passed				

5.8 Performance and Scalability

Test ID	041	**Category**	Performance	**Severity**	High
Objective	Ensure that passport scanning is fast and efficient under various conditions.				
Steps	1. Scan a passport under optimal lighting conditions. 2. Measure the time taken from scan initiation to data extraction. 3. Repeat the test under different lighting and angles.				
Expected	Passport scanning completes within an acceptable time. The system provides consistent performance under different conditions. Poor lighting or angles do not cause excessive delays.				
Date-Result	01/05/2025 - Passed				

Test ID	042	**Category**	Scalability	**Severity**	High
Objective	Verify that storing and managing a large number of commitments does not degrade app performance.				
Steps	1. Store an increasing number of identity proofs. 2. Measure app launch time and navigation responsiveness. 3. Open and view multiple stored proofs in succession. 4. Observe memory and CPU usage during operations.				
Expected	The app remains responsive even with a large number of stored proofs. No noticeable slowdown in UI responsiveness. Memory and CPU usage remain within acceptable limits.				
Date-Result	01/05/2025 - Passed				

Test ID	043	**Category**	Performance	**Severity**	High
Objective	Ensure that blockchain verification costs are optimized.				
Steps	1. Initiate commitment CID verification via blockchain. 2. Measure gas fees and transaction costs for different operations. 3. Optimize smart contract execution to reduce computation overhead.				
Expected	Transaction costs are minimized without sacrificing security. Gas fees remain predictable and within acceptable limits.				
Date-Result	01/05/2025 - Passed				

Test ID	044	**Category**	Performance	**Severity**	Critical
Objective	Ensure that identity proof verification is fast and efficient.				
Steps	1. Submit an identity proof for verification. 2. Measure the time taken to complete verification. 3. Optimize cryptographic operations and proof validation logic.				
Expected	Verification completes within an acceptable time. Optimization techniques improve speed.				
Date-Result	01/05/2025 - Passed				

5.9 Usability and User Experience

Test ID	045	**Category**	Usability	**Severity**	High
Objective	Ensure that the mobile app UI is clear, easy to navigate, and user-friendly.				
Steps	1. Launch the mobile app and navigate through all main screens. 2. Check the layout, button placements, and visual hierarchy. 3. Test common actions like scanning a passport, verifying an identity, and managing proofs. 4. Check the information density throughout these actions.				
Expected	Navigation is smooth, with clear labels and icons.				
Date-Result	01/05/2025 - Passed				

Test ID	046	**Category**	Usability	**Severity**	High
Objective	Verify that users can easily use the browser extension for identity verification.				
Steps	1. Install and enable the browser extension. 2. Attempt an identity verification request from a website. 3. Ensure that the request is clearly displayed in the extension. 4. Approve/reject the request and observe the workflow.				
Expected	Users can easily understand and complete identity verification. The extension provides a simple and clear interface. Key actions are accessible and well-explained.				
Date-Result	01/05/2025 - Passed				

Test ID	047	**Category**	UX	**Severity**	High
Objective	Ensure that users receive informative and actionable feedback during identity verification.				
Steps	1. Initiate an identity verification request. 2. Observe feedback messages for success, failure, and pending states. 3. Check if error messages are clear and helpful. 4. Verify that loading indicators and progress messages are present.				

Expected	Users receive clear feedback for all actions. Success and failure messages provide meaningful details. The app avoids technical jargon and uses user-friendly language.
Date-Result	01/05/2025 - Passed

Test ID	048	**Category**	UX	**Severity**	Critical
Objective	Ensure that user mistakes are handled smoothly without frustration.				
Steps	1. Enter an incorrect PIN multiple times and observe the response. 2. Try scanning an invalid or damaged passport. 3. Submit an incorrect identity proof for verification. 4. Observe how the system guides the user to correct mistakes.				
Expected	The system provides helpful guidance. Users are not locked out or forced to restart unnecessarily. The app prevents excessive retries for security-sensitive actions.				
Date-Result	01/05/2025 - Passed				

Test ID	049	**Category**	Accessibility	**Severity**	High
Objective	Ensure that visually impaired users can navigate and use the app effectively.				
Steps	1. Enable screen reader functionality. 2. Navigate through the app using only voice commands or accessibility tools. 3. Verify that all essential actions can be performed without visual input. 4. Check for proper contrast ratios, readable font sizes, and alt text for images.				
Expected	The app is fully accessible via screen readers. All UI elements are properly labeled for assistive technologies. Text and buttons maintain adequate visibility and contrast.				
Date-Result	01/05/2025 - Passed				

Test ID	050	**Category**	UX	**Severity**	High
Objective	Ensure that first-time users can easily understand and set up the app without confusion.				

Steps	1. Install the mobile app and launch it for the first time. 2. Follow the onboarding process, including setting up authentication and scanning the first identity proof. 3. Observe the clarity of instructional messages and tooltips. 4. Check if users can skip onboarding and access a help section later.
Expected	The onboarding process is clear and guides users smoothly. Instructions are simple and informative. Users can complete the initial setup without external assistance.
Date-Result	01/05/2025 - Passed

6. Maintenance Plan and Details

Maintaining the IAWIA system is crucial to ensuring its continued reliability, security, and effectiveness. The following sections outline how different parts of the system can be updated or maintained, the challenges we might face in the future, and how we will address issues like bugs, security patches, and user feedback.

6.1 Mobile App Maintenance

The mobile app is a core component that needs frequent maintenance to stay compatible with updates to mobile operating systems and ensure that cryptographic protocols remain secure. Maintenance tasks include:

Updates and Bug Fixes: Regular updates will address performance issues, update libraries, and fix bugs. We will use user feedback to prioritize issues and improvements.

Cryptographic Library Updates: Libraries such as circomlib or snarkjs must be updated if vulnerabilities are found or if there are new versions with better performance or security.

OS Compatibility: As Android and iOS receive updates, we will ensure that the app remains functional with the latest OS features and security policies.

6.2 Browser Extension Maintenance

The browser extension allows users to manage their identities securely. To ensure the wallet works effectively:

Bug Fixes: Immediate updates will be made to address critical bugs reported by users or detected during testing.

Security Patches: Since the extension handles sensitive data, it will undergo regular security audits, and vulnerabilities will be patched promptly.

User Feedback: Regular UI improvements and the addition of new features based on user feedback will be made to improve usability and performance.

6.3 Backend Server Maintenance

The backend server is a crucial component for 3rd party clients to interact with the IAWIA system. We will maintain the API with:

API and ABI compatibility: We will use proper versioning schemes and public documentation to communicate API and ABI changes and backward compatibility commitments.

Security Audits: All arithmetic circuits will undergo periodic audits to identify and fix vulnerabilities before they cause any harm.

6.4 Bug Fixes, Security Patches, and User Feedback

Bug Tracking: We will use platforms like GitHub to track and manage bugs. Critical issues will be handled immediately, while less important ones will be addressed in regular updates.

Security Patches: Security vulnerabilities will be treated as a priority. Once identified, patches will be deployed swiftly.

User Feedback: We will integrate feedback channels within the app and extension to ensure that users' concerns and suggestions are considered in future updates.

6.5 Future Challenges

New Cryptographic Risks: Advances in quantum computing or cryptographic attacks may require changes in the zero-knowledge proof protocols. We will stay updated on cryptographic research and adapt when necessary.

Regulatory Changes: As regulations around digital identity evolve, we will update the system to stay compliant with new laws like GDPR or eIDAS 2.0.

Blockchain Evolution: Changes in Ethereum or new Layer-2 solutions may require adjustments to our system to maintain efficiency and reduce costs.

7. Other Project Elements

7.1 Consideration of Various Factors in Engineering Design

The engineering design of the IAWIA project has been influenced by multiple factors, from technical requirements to security measures, economic considerations, and ethical constraints. Each of these factors has played a significant role in shaping the system's architecture, implementation, and overall design decisions. The following sections further expand on the constraints that influenced the development process and the standards adhered to throughout the project.

7.1.1 Constraints

Throughout the development of the IAWIA project, several constraints impacted the design and implementation of the system. These constraints were considered carefully to ensure the project remained feasible, secure, and efficient. Below are the key constraints that affected the project during its development:

Technical Constraints

Browser Compatibility: One of the first technical limitations encountered was ensuring compatibility with various browsers. The system's Google Chrome extension was specifically designed to be compatible with Chrome-based browsers, such as Google Chrome, Microsoft Edge, and Brave Browser. However, the integration with other browsers, like Firefox or Safari, was not initially considered due to the added development time and complexity required. This constraint meant that the system's users were limited to Chrome-based browsers for the initial version of the product.

Wallet Integration: Ensuring seamless and secure interactions between the browser extension, mobile app, and 3rd party clients required careful planning and implementation. The wallet integration had to be highly secure to protect users' identity and personal information.

Zero-Knowledge Proof (ZKP) Computation Time: Implementing zero-knowledge proofs (ZKPs) on mobile devices and web browsers proved challenging due to the computational expense of these cryptographic protocols. As ZKPs are necessary for maintaining user privacy while verifying identity, it was essential to optimize the computation process. This constraint required us to pivot to off-chain computations.

Security and Privacy Constraints

No Centralized Data Storage: A core principle of the project was that no sensitive user data would be stored on a central server. All data would be stored locally on the user's device and encrypted on the blockchain. This constraint eliminated traditional centralized storage solutions, making it more challenging to implement certain features, such as data retrieval and user data management. The trade-off was improved privacy and security but with added complexity in handling data locally.

Authentication and Authorization: The system's authentication model, based on passport verification and zero-knowledge proofs (ZKPs), presented its own set of security challenges. Ensuring that authentication tokens were stored temporarily in the browser's memory required extra measures to safeguard against unauthorized access and minimize the risk of data exposure [11].

Attack Vectors: Browser extensions, by nature, are vulnerable to attacks such as Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF), and Man-in-the-Middle (MITM) attacks. To mitigate these risks, we implemented robust security measures such as using HTTPS for communication, secure Web3 APIs, and operating in secure sandbox environments for sensitive operations. These security constraints had to be incorporated into the design from the outset.

Economic Constraints

Gas Fees: The use of blockchain for identity verification and other processes inherently introduces gas fees, especially on Ethereum and Layer-2 networks. This became an important economic constraint, as users were required to pay gas fees for actions like identity verification. The team had to carefully evaluate the cost-benefit ratio for users and ensure that the system's overall utility justified these fees.

Development and Maintenance Costs: The creation and maintenance of the system's various components—such as the Chrome Extension, mobile app, and backend server—introduced ongoing costs [12]. These costs include hosting services, server security, user support, and regular updates. Balancing these costs with the project's potential to be sustainable in the long term was an important consideration during development.

Ethical Constraints

User Anonymity: The system was built to ensure that users' identities remained private and anonymous. No personally identifiable information (PII) would be tracked or stored by the system. This ethical constraint influenced many decisions, particularly when designing how user data would be encrypted and processed. We ensured that users could control their personal data, allowing them to participate in identity verification without compromising their anonymity.

Legal Compliance: The project needed to adhere to global data protection regulations, especially the General Data Protection Regulation (GDPR) in the European Union. This constraint required that the system be designed with user privacy at the forefront, implementing features such as the ability for users to manage, delete, or export their personal data. The legal requirement for compliance added complexity to the system's design but was essential for ensuring that the system met international standards for data protection and privacy.

7.1.2 Standards

In addition to addressing constraints, the project followed several industry standards to ensure the system's security, interoperability, and compliance. These standards were incorporated into both the design and development phases of the IAWIA project:

Security Standards

X.509 Public Key Infrastructure (PKI): Digital certificates and encryption mechanisms based on X.509 PKI standards were used to validate and verify the identity of users. This standard ensured that the system could rely on trusted, certificate-based identity management for secure communications.

Blockchain and Cryptography Standards

SHA-512 and ECDSA secp521r1 Hash Algorithms: The system relied on the SHA-512 and ECDSA secp521r1 hashing algorithms for data integrity and cryptographic security. These hashing algorithms ensured that sensitive data, including user identities, were securely stored and transmitted.

Zero-Knowledge Proof (ZKP) Standards: The ZKP protocol used in the system adhered to industry-recognized standards such as zk-SNARKs to ensure that users could prove their identity without revealing any personal information [13].

Software and Development Standards

UML 2.5.1: The system architecture and components were modeled using UML 2.5.1 diagrams to provide a clear and standardized representation of the system design. These diagrams helped developers and stakeholders understand the interactions between various components.

Circom and snarkjs: Cryptographic operations were managed using libraries like circomlib and snarkjs, which are standard for interacting with zero-knowledge protocols. These libraries ensured that the system could efficiently handle blockchain transactions.

User Experience and Accessibility Standards

WCAG 2.1 (Web Content Accessibility Guidelines): The user interface was designed to be fully accessible for individuals with disabilities, following WCAG 2.1 guidelines to ensure that all users, regardless of ability, could access and use the system.

Material Design Guidelines: The UI components followed Google's Material Design principles to ensure consistency, usability, and an intuitive user experience.

7.2 Ethics and Professional Responsibilities

During the development of the IAWIA project, a strong emphasis was placed on ethical principles, particularly when dealing with sensitive user data such as passports and personal identity information. Ensuring privacy, security, and non-discrimination were critical in the design and implementation phases, as the system is directly involved with personal and confidential information.

Handling Sensitive User Data

Decentralized Data Storage: IAWIA ensures that user data, including sensitive personal details, is never stored on centralized servers. Instead, it uses a decentralized approach where all data is securely stored locally on the user's device and is encrypted before being stored on IPFS. This approach minimizes the risk of large-scale data breaches.

Zero-Knowledge Proofs (ZKPs): To further enhance privacy, the system employs ZKPs to validate the identity of the user without revealing any sensitive information. This ensures that personal details like name, date of birth, and passport numbers are never exposed during verification [14].

Encryption Protocols: All communication and transactions within the system are encrypted using HTTPS, ensuring that data transmitted over the network is secure and protected from potential attacks or interception.

Ensuring Security, Privacy, and Non-Discrimination

User Privacy: The IAWIA system collects only the essential data needed for identity verification. By adopting a minimalist approach, the system limits the amount of personally identifiable information shared, thereby reducing the potential for misuse.

Non-Discrimination: The platform is designed to be inclusive, allowing all users—regardless of their nationality, ethnicity, or background—to authenticate and prove

their identity. The system was carefully developed to avoid any form of discrimination in the verification process.

GDPR Compliance: The project adheres to the General Data Protection Regulation (GDPR) guidelines, ensuring that all users' personal information is handled in a lawful, transparent, and secure manner. Users have the right to access, modify, or delete their data whenever they choose, empowering them with full control over their personal information.

Professional Work Ethics

Punctuality: Throughout the project, we ensured that all tasks were completed on time, meeting deadlines and keeping the project moving forward. This helped in maintaining momentum and allowed for timely reviews and iterations.

Collaboration and Respect for Teammates: We maintained an open line of communication with all team members. Each member's ideas and contributions were valued, and we made sure to work collaboratively, ensuring that everyone's perspective was considered in the decision-making process.

Accountability: We took full responsibility for our work and ensured that we communicated any challenges we faced. If any errors occurred, we took steps to correct them promptly and learned from the experience to prevent similar issues in the future.

Ethical Coding Practices: In the development process, we adhered to ethical coding standards, ensuring the software was secure, functional, and compliant with industry regulations. We avoided practices that could compromise user security or privacy, such as leaving hardcoded credentials or failing to secure sensitive data properly.

In summary, throughout the IAWIA project, we were committed to maintaining the highest ethical standards. This involved protecting user privacy, ensuring data security, and respecting all individuals regardless of their background. By aligning the project with ethical guidelines, We were able to contribute to a system that is not only secure but also trustworthy and reliable.

7.3 Teamwork Details

7.3.1 Contributing and Functioning Effectively on the Team

For the success of the IAWIA project, it is essential that every team member actively contributes and takes responsibility for assigned tasks. We aim to distribute the workload evenly so everyone can focus on specific components and ensure the system's overall quality. To achieve this, we follow a structured workflow that includes clear task assignments, regular meetings, and collaborative development processes.

We use GitHub to track contributions and ensure that all members participate in coding, debugging, and reviewing. Each member commits their changes to the repository, and code reviews are conducted to maintain high-quality standards. Additionally, we follow agile principles, dividing tasks into smaller, manageable units, allowing us to progress steadily. By documenting our development process and sharing technical knowledge, we ensure no team member is left behind and continuously support each other to overcome challenges.

We perform regular evaluations at the end of each sprint to maintain accountability. This helps us identify areas for improvement and ensure that everyone is effectively contributing. We encourage team members to voice their concerns and suggestions, fostering a culture of continuous learning and improvement.

7.3.1.1 Emrehan Ateş

I contributed to the project by focusing on generating ZKPs and their utilization on the blockchain. My efforts included local SNARK generation and developing strategies for storing these proofs on-chain and off-chain. I consistently collaborated with my teammates to effectively integrate ZKPs with the wallet. Additionally, I played a key role in implementing IPFS into IAWIA and worked on leveraging our SNARKs for on-chain verification through smart contracts and zero-knowledge virtual machines. I also actively participate in group meetings to introduce effectiveness and novelty to our project idea.

7.3.1.2 Eren Karakaş

I mainly focused on the cryptographic procedures in the project, notably generating the ZKPs and constructing the necessary circuits. Together with Emrehan, we deliberated on the implementation details of the IAWIA system's cryptographic backend, such as where to store the information, how to generate and store the proofs, and other similar considerations. I have also implemented the backend server IAWIA utilizes. I made a deliberate effort to maintain our vision and keep the features grounded during our meetings. Additionally, I contribute considerably to our reports.

7.3.1.3 Fırat Utku Gül

I actively contributed to developing the IAWIA system by focusing on the mobile application. I worked on implementing key functionalities, including passport scanning via NFC and secure local storage of identity attributes. Additionally, I collaborated with the team to optimize the user experience, ensuring the app provides a seamless and intuitive interface for identity management. I also participated in debugging, testing, and refining the mobile application to enhance its performance and reliability. Beyond development, I contributed to documentation, helping maintain clear technical guidelines for future improvements.

7.3.1.4 Mehmet Emre Güneş

I contributed to preparing reports throughout the project process and ensured that the documentation was complete and organized. I also conducted market research to analyze the

potential areas of use of our project and made comparisons with existing solutions. These analyses gave our team essential insights into how our project could be positioned in the sector. In technology selection, I informed my teammates by evaluating the applicability of the tools and systems used and helped determine the most appropriate technologies for the project.

7.3.1.5 Serhat Merak

I contributed to the design and development of the Chrome extension, including encrypted data transfer through mobile applications and extensions. Also, I am actively developing the verification system and communication between the external website and our extension. Besides development, I actively participated in team meetings and contributed to the idea of IAWIA and how it should work.

7.3.2 Helping Creating a Collaborative and Inclusive Environment

We believe that a strong team environment is essential for productivity and motivation. Therefore, we prioritize open communication, respect, and inclusivity in our teamwork. Every team member's opinion is valued, and decisions are made collectively through discussions. We ensure that all members have equal opportunities to contribute, regardless of their experience level.

To promote collaboration, we hold weekly face-to-face or online meetings where we discuss progress, challenges, and upcoming tasks. We conduct short online Zoom meetings to address urgent matters and ensure alignment. These meetings allow team members to share updates, ask questions, and seek assistance when needed.

We also use platforms like Google Docs for documentation sharing, ensuring that all relevant information is accessible to every team member. By maintaining detailed records of discussions, technical challenges, and solutions, we create a knowledge base that benefits the entire team. Pair programming is encouraged during critical development stages, enabling knowledge transfer and improving code quality.

Furthermore, we actively support and motivate each other. If a team member faces difficulties, others step in to assist, ensuring that no one feels isolated. By fostering a collaborative and inclusive environment, we enhance team synergy and achieve better results in our project.

7.3.2.1 Emrehan Ateş

I consistently sought input from my teammates when making critical project decisions, such as updating the tech stack or altering functional requirements. Fostering an inclusive environment was our priority, so we revised our project concept throughout the first semester to ensure it remained engaging and enjoyable for everyone involved. Each team member has their own strengths, as do I. We shared our new insights with the team, even though it wasn’t

necessarily anyone's responsibility to transform this project into an educational experience. I actively used our communication channels to keep my peers updated.

7.3.2.2 Eren Karakaş

I regularly share updates on our decisions and challenges related to cryptography with the entire team. We continuously review our project's design and implementation ideas to ensure that everyone's concerns are addressed. We make it a priority to share our insights as a team so everyone stays informed about various decisions, even if the decision would not directly impact their area of work. Additionally, I communicate all my decisions regarding report edits or correspondence with instructors and await confirmation before moving forward.

7.3.2.3 Fırat Utku Gül

To foster a collaborative and efficient working environment, I actively contributed to team discussions and problem-solving sessions, ensuring that technical challenges were addressed collectively. I regularly shared updates on the application's progress, highlighting key developments and potential improvements. Additionally, I assisted my teammates by troubleshooting issues and providing insights into implementation details. I also documented important aspects of the development process, making it easier for the team to track progress and maintain clarity in our workflow. I helped create an inclusive and productive team environment by maintaining open communication and offering support when needed.

7.3.2.4 Mehmet Emre Güneş

In order to increase collaboration within the team and ensure that everyone has easy access to information, I organized the project documentation to ensure that the shared information was understandable and accessible. In addition, I regularly shared my research and findings with my teammates to facilitate the team's access to technical issues. I contributed to the creation of a platform where our team can follow the progress and developments by taking care of the project's website.

7.3.2.5 Serhat Merak

In order to improve collaboration within the team, I organized the team meetings and ensured that every one of my teammates was actively participating in the decisions given throughout the project development. Also, by encouraging my teammates to share their project's blockers and current status, I supported the distribution of the information between my teammates.

7.3.3 Taking Lead Role and Sharing Leadership on the Team

Leadership in our team is not limited to one person; instead, we embrace a shared leadership model where responsibilities are distributed among team members. Each member takes ownership of a specific area of the project and ensures that it progresses efficiently. By

distributing leadership responsibilities, we create a balanced workload and encourage everyone to take initiative.

Team members lead in different aspects, such as technical decision-making, task management, and problem-solving. Those with expertise in specific areas provide guidance to others, ensuring that knowledge is effectively shared. Leadership roles rotate depending on the project's needs, allowing everyone to develop leadership skills and contribute meaningfully to the team’s success.

We also encourage proactive behavior, where team members suggest improvements, propose solutions to challenges, and take the initiative to implement new ideas. By allowing everyone to lead at different times, we create a dynamic team structure where responsibilities are not placed on a single individual but shared for maximum efficiency.

In addition, we regularly evaluate leadership effectiveness and make necessary adjustments to improve team coordination. By maintaining an open feedback loop, we ensure that leadership within the team is flexible and adaptive to our project's evolving needs.

Through this approach, we cultivate a team culture that values responsibility, proactive engagement, and mutual respect, which ultimately contributes to the overall success of the IAWIA project.

7.3.3.1 Emrehan Ateş

I was responsible for the ZK technology stack within the project. I researched and evaluated the proof generation tools, learning how to implement them effectively. My primary collaboration was with Eren, and we operated without designated leaders. As this was our first experience with ZK, we shared our insights and combined our knowledge to pave the way forward. Additionally, I took the lead in presenting last semester. I assisted the team in allocating topics based on each member's strengths and weaknesses and was responsible for ensuring the presentation's coherence and cohesion.

7.3.3.2 Eren Karakaş

Emrehan and I share the leadership in exploring ZK technology. I have researched potential methods for implementing our circuits, extracting passport information, generating ZK proofs, and created proof-of-concept implementations for these tasks. We have consistently shared our findings and aimed to decrease the complexities of the zero-knowledge-proof field. Additionally, I take the lead in finalizing our reports to ensure they are coherent and complete.

7.3.3.3 Fırat Utku Gül

I took the lead in the development of the mobile application, ensuring that the implementation aligned with the project’s goals and technical requirements. I was responsible for researching and integrating key technologies needed for mobile functionality, particularly focusing on NFC scanning and cryptographic proof generation. By addressing technical

challenges, I helped streamline the development process. Additionally, I actively supported my teammates by providing guidance on development issues and troubleshooting implementation challenges. I contributed to technical decision-making, ensuring that the mobile app was both secure and user-friendly. Through continuous collaboration and knowledge sharing, I played a key role in keeping the mobile development on track and ensuring its successful integration into the overall IAWIA system.

7.3.3.4 Mehmet Emre Güneş

I led the process of determining applicable technologies by researching the project's technical aspects. In particular, I analyzed whether the technologies to be selected were suitable for the project requirements and enabled the team to make informed decisions. In addition, I actively participated in the writing and editing of the reports that were critical to the project, ensuring that the documentation was completed in a complete and professional manner. In addition, I was responsible for managing the project's website, organizing the team's progress, and making the necessary arrangements for the project to be introduced to the outside world.

7.3.3.5 Serhat Merak

I led the process of deciding the phases of the application based on different deadlines. Every month, I led the decision of how many different branches of our project should be developed and communicated. For example, in the first phase of our project, ZKPs were produced and stored in the mobile app. For the second phase, we decided that ZKPs should be stored in IPFS, besides the hashed personal unique data. The last phase is where all the ZKPs and hashed unique data are stored in the blockchain. I was responsible for managing the time and comparing where we were and where we should be.

7.3.4 Meeting Objectives

At the start of the IAWIA project, we outlined a series of objectives and milestones in our project plan. Reflecting on the progress, we can compare our initial goals with what was actually achieved.

Planned Objectives:

Develop a Decentralized Identity System: Our primary objective was to create a decentralized identity verification system that provides users with control over their personal data. We set out to integrate blockchain technology and zero-knowledge proofs (ZKPs) to ensure privacy and security.
Create a Mobile Application and Web Extension: We planned to build a mobile app and a browser extension that would interact with the decentralized system. The mobile app was intended to facilitate identity verification, while the web extension would serve as a crypto wallet and handle transactions securely.

Integrate Blockchain and Smart Contracts: Another key objective was to implement smart contracts on the blockchain to automate identity verification and ensure the integrity of data. This would enable users to prove their identity without disclosing sensitive information.

Achieved Objectives:

Decentralized Identity System: We successfully developed and implemented a decentralized identity verification system. By leveraging blockchain and ZKPs, we ensured that users could verify their identity without revealing personal information. This was fully achieved and forms the core of our solution.

Mobile Application and Web Extension: The mobile app was developed successfully, providing users with a secure platform for identity verification. The web extension was also completed and allows users to interact with the system through a browser using a crypto wallet. While both the app and extension met their primary goals, some minor optimizations in user experience and performance are still in progress and may require further attention in future updates.

Unachieved Objectives:

Blockchain and Smart Contracts: We did not implement Ethereum-based smart contracts for identity verification. Certain technical challenges, such as gas fee costs and transaction speeds on the blockchain, were not fully resolved during the project timeline. Additionally, they came with performance problems that made mobile application usage hard. Therefore, we decided to use off-chain verification instead.

Cross-Browser Compatibility for the Web Extension: We planned to extend the web extension's compatibility to browsers beyond Chrome, such as Firefox and Safari. However, we faced time constraints and could not complete this integration. We intend to revisit this after the project's current phase.

Overall, we made significant progress toward meeting our objectives. While some challenges remained unresolved, we accomplished the most critical goals, including the development of a decentralized identity system and the successful implementation of blockchain-based smart contracts. The remaining issues, such as on-chain verification being unavailable and incomplete browser compatibility, are areas that will be addressed in future iterations of the project.

7.4 New Knowledge Acquired and Applied

During the development of the IAWIA project, our team gained valuable insights and expanded our technical knowledge in several key areas, including zero-knowledge proofs (ZKP), decentralized identity systems, and IPFS implementations [15].

Zero-Knowledge Proofs (ZKP): We explored the concept of zero-knowledge proofs, which formed the foundation of the system's privacy features. ZKPs allow users to verify their identity without revealing any sensitive data, which is crucial for maintaining privacy in identity verification. As we implemented ZKP solutions for our mobile app, we learned how computationally expensive these proofs can be, especially on mobile devices [16]. Additionally, since ZKP technology is extremely new, we have reached out to other project developers to gain practical knowledge on real-world usage.

Decentralized Identity Systems: Working with decentralized identity (DID) solutions was another key aspect of the project. We learned about the potential advantages of decentralized systems in terms of security and user control over personal data. By leveraging blockchain and IPFS, our team was able to eliminate the need for centralized data storage, which significantly reduced the risks of data breaches and hacks [17]. This shift from traditional centralized systems to decentralized approaches was a major takeaway and influenced the design and architecture of our solution.

IPFS Implementation: We used IPFS to store ZKP commitments for later use. We learned about decentralized databases and filesystems. We first thought of using one of our old computers as a node in the IPFS. However, we decided not to do that and rented a node in the system instead. We learned how to push, delete, and traverse data in the IPFS protocol.

In summary, the IAWIA project provided our team with a solid understanding of blockchain, cryptography, and decentralized identity systems. The knowledge we gained was not only theoretical but was also directly applied in the development of the project, allowing us to build a secure and privacy-preserving identity verification system.

8. Conclusion and Future Work

In conclusion, our system has successfully developed a decentralized identity verification solution, addressing key issues with traditional centralized identity systems. By leveraging blockchain technology, zero-knowledge proofs (ZKPs), and a seamless user experience through a mobile app and browser extension, we have created a secure and privacy-preserving platform that puts users in control of their identity data. The system ensures that no sensitive data is stored centrally, protecting against potential security breaches common in centralized systems.

However, we acknowledge that the current implementation has room for improvement. As the adoption of decentralized identity solutions grows, we see opportunities for scaling the system to handle larger user bases and support on-chain implementations. One of the main areas for future work is trying to provide acceptable performance for on-chain verification and provide it as an alternative to off-chain verification.

Additionally, expanding the system's compatibility with a broader range of blockchain platforms, as well as incorporating more sophisticated cryptographic techniques, will be a key focus in our future development efforts. This will allow us to offer more robust solutions for different use cases and ensure that the system remains adaptable to changing technologies and regulations.

In the future, our goal is to continue enhancing the platform's capabilities, refining the user experience, and staying at the forefront of developments in decentralized identity management.

9. Glossary

Blockchain: A distributed information technology where transactions are verified and stored without the need for a central authority.

Zero-Knowledge Proof (ZKP): A cryptographic method that allows a user to prove a claim without revealing any underlying information.

Circom: A domain-specific language for creating arithmetic circuits, primarily used in zero-knowledge proof systems.

SnarkJS A JavaScript library that allows you to generate, verify, and interact with zk-SNARK proofs created with Circom circuits.

Ethereum: A popular blockchain platform that supports smart contracts.

NFC (Near Field Communication): A short-range wireless communication protocol commonly used for credit cards and ID cards.

IPFS (InterPlanetary File System): A decentralized file system for storing and managing files and data. Unlike traditional storage, it allows data to be stored and published in a peer-to-peer fashion.

CID (Content Identifier): A unique identifier that is assigned to each piece of data stored in the IPFS. It represents the address of that piece of data and points to its location.

Know Your Customer (KYC): A process used by businesses to confirm the identity of their customers, typically for compliance and security purposes.

Decentralized Identifier (DID): A unique identifier that allows individuals to manage their own identity without relying on a central authority.

Public Key Infrastructure (PKI): A security framework that helps with safe communication and authentication using cryptographic keys.

Passport Scanning Service: A service in the IAWIA system that enables users to extract identity attributes from their passports using NFC technology.

Zero-Knowledge Proof Generation Service: A service that converts identity attributes into cryptographic proofs, enabling verification of specific claims without revealing private data.

Browser Extension Authentication Service: A service that enables users to verify their identity while accessing online services without sharing personal data.

Cryptographic Commitments: Hashes or cryptographic representations stored on a blockchain that are used to verify proofs without exposing sensitive data.

End-to-End Encryption: A method of encrypting data so that only the communicating parties can access it, preventing interception by third parties.

Secure Key Management: The process of securely storing and handling cryptographic keys to ensure data privacy and security.

Selective Disclosure: A privacy-enhancing feature that allows users to reveal only the necessary identity attributes while keeping others hidden.

Secure Storage Mechanisms: Various encrypted storage solutions are used on mobile devices, such as Secure Enclave on iOS and Keystore on Android, to protect sensitive identity data.

10. References

1. M. Maureen, “Blockchain identity management: A complete guide,” 1Kosmos, Feb. 14, 2025. [Online]. Available: https://www.1kosmos.com/blockchain/blockchain-identity-management-a-complete-guide
2. “Decentralized identity: The ultimate guide 2025,” Dock.io. [Online]. Available: https://www.dock.io/post/decentralized-identity
3. R. Adhikary, “Centralized vs decentralized identity management,” CloudEagle, Oct. 07, 2024. [Online]. Available: https://www.cloudeagle.ai/blogs/centralized-vs-decentralized-identity-management
4. “Zero-knowledge proofs: A beginner’s guide,” Dock.io. [Online]. Available: https://www.dock.io/post/zero-knowledge-proofs#:~:text=The%20paper's%20definition%20of%20a,developed%20from%20being%20a%20purely
5. “How a centralized database makes a difference?,” Qualityze, Aug. 01, 2023. [Online]. Available: https://www.qualityze.com/blogs/how-a-centralized-database-makes-a-difference
6. “EKYC explained: Why the future is digital,” Regula, Jan. 23, 2025. [Online]. Available: https://regulaforensics.com/blog/what-is-ekyc/
7. L. Hendrickson, “Preventing data breaches with decentralized identity,” Identity, Jan. 21, 2025. [Online]. Available: https://www.identity.com/preventing-data-breaches-with-decentralized-identity/
8. C. Mazzocca, A. Acar, S. Uluagac, P. Bellavista, and M. Conti, “A survey on decentralized identifiers and verifiable credentials,” 2024.
9. S. R, R. C. Nair, and P. K. Panakalapati, “Promise of zero‐knowledge proofs (ZKPs) for blockchain privacy and security: Opportunities, challenges, and future directions,” Security and Privacy, Sep. 2024. [Online]. Available: doi: 10.1002/spy2.461
10. “Use cases and requirements for decentralized identifiers,” W3C, Mar. 17, 2021. [Online]. Available: https://www.w3.org/TR/did-use-cases/
11. S. Y. Lim, P. Fotsing, A. Almasri, O. Musa, M. L. Kiah, T. Ang, and R. Ismail, “Blockchain technology the identity management and authentication service disruptor: A survey,” International Journal on Advanced Science, Engineering and Information Technology, vol. 8, p. 1735, 2018. [Online]. Available: doi: 10.18517/ijaseit.8.4-2.6838
12. T.-T. Kuo, H. Rojas, and L. Ohno-Machado, “Comparison of blockchain platforms: A systematic review and healthcare examples,” Journal of the American Medical Informatics Association, vol. 26, 2019. [Online]. Available: doi: 10.1093/jamia/ocy185
13. S. Almuhammadi, “One-round zero-knowledge proofs and their applications in cryptographic systems,” Ph.D. dissertation, University of Southern California, USA, 2005. [Online]. Available: Order Number: AAI3180486
14. “EPassport basics,” ICAO. [Online]. Available: https://www.icao.int/Security/FAL/PKD/Pages/ePassport-Basics.aspx

15. “A survey on the applications of zero-knowledge proofs,” arXiv. [Online]. Available: https://arxiv.org/html/2408.00243v1
16. Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology, U.S. Department of Commerce, “TLS guidelines: NIST publishes SP 800-52 Revision 2,” CSRC. [Online]. Available: https://csrc.nist.gov/news/2019/nist-publishes-sp-800-52-revision-2
17. Š. Čučko and M. Turkanović, “Decentralized and self-sovereign identity: Systematic mapping study,” IEEE Access, 2021. [Online]. Available: doi: 10.1109/ACCESS.2021.3117588

11. User Manual for IAWIA

Introduction

IAWIA is a decentralized identity verification system that combines a web browser extension and a mobile application. It allows users to manage their digital identities securely, privately, and independently using blockchain technology and zero-knowledge proofs. With IAWIA, users can verify their identity without revealing unnecessary personal data and manage encrypted identity proofs tied to their wallets.

System Requirements

Web Extension

- Compatible with Chrome-based browsers (Google Chrome, Brave, Microsoft Edge)
- Internet connection

Mobile Application

- Android smartphone with NFC capability
- Internet access
- Android 8.0 or higher (recommended)

Installation Instructions

Web Extension

1. Download the extension from the Chrome Web Store or install it manually via developer mode.

2. Click "Add to Chrome".

3. Pin the extension to your browser for easy access.

Mobile Application

1. Install the APK file manually or from the extension redirect.

2. Grant necessary permissions such as Camera, NFC, and Storage.

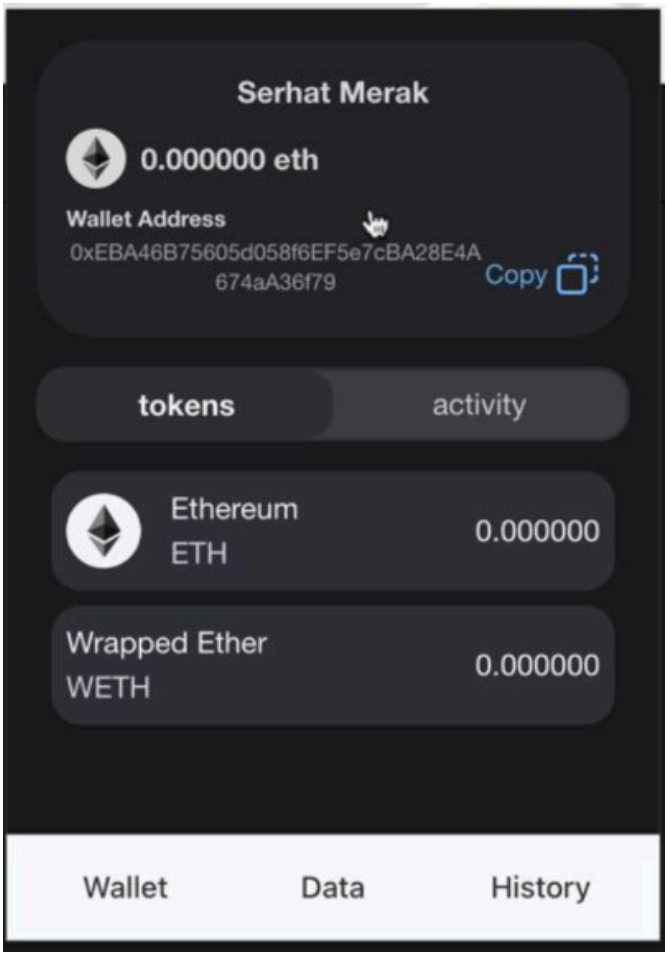
3. Launch the app after installation.

Using the Web Extension Wallet

1. Launch the extension.

2. Import your wallet using a wallet key or seed phrase.

3. After importing, you can view your Ethereum wallet address, ETH balance, Wrapped Ether (WETH) balance, Activity, and History.



Identity Verification with the Extension

1. Click on the relevant component on the web page.

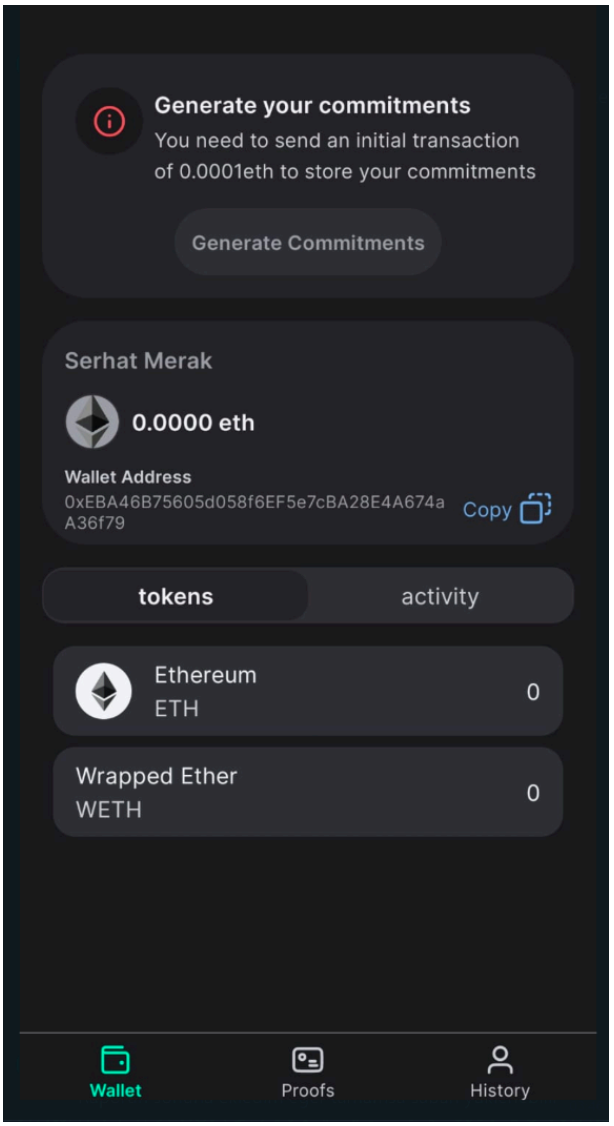
2. See the identity attributes the service wants you to disclose.

3. Authorize the web extension to disclose the necessary information.

Using the Mobile Application

Dashboard

View your Ethereum wallet balance and address, list of identity proofs, and stored commitments.



Exporting Proof to Wallet

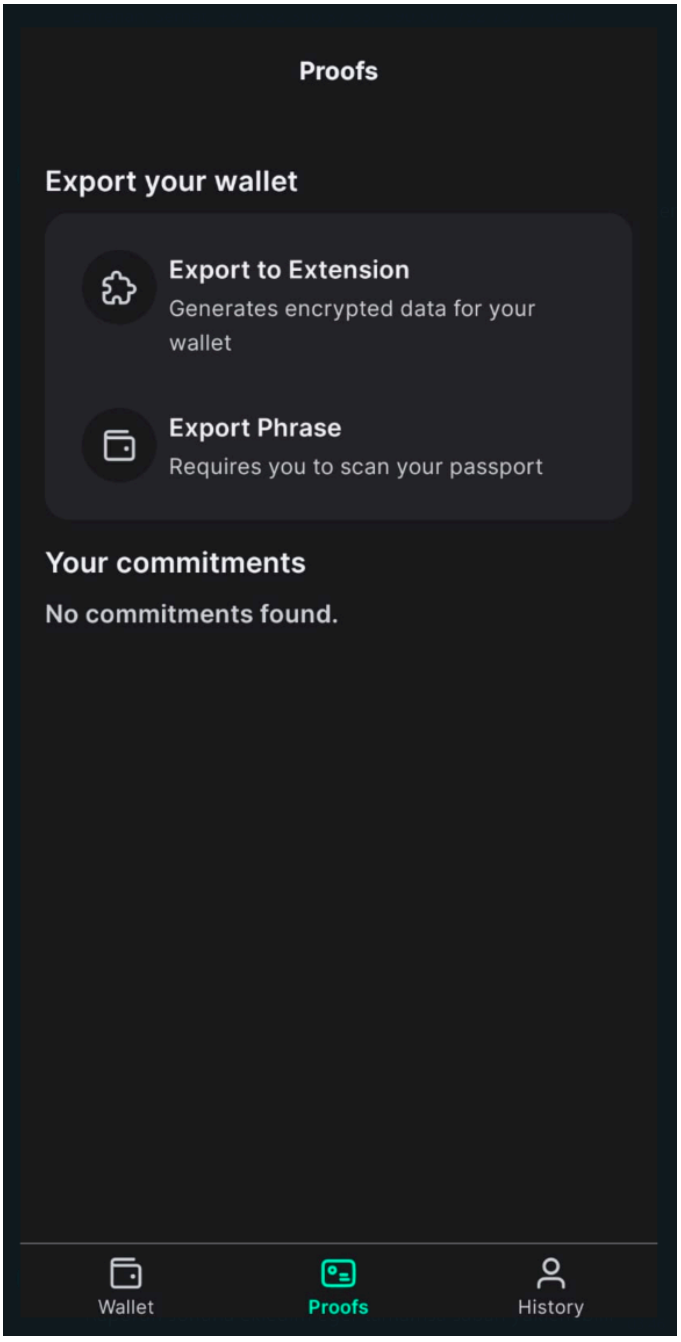
Tap "Export to Extension".

This generates the encrypted data and secrets that can be sent to the wallet for later verification requests.

Export Security Phrase

Tap "Export Phrase".

Requires passport verification through camera and NFC scanning.



Commitments

Commitments are cryptographic proofs linked to your wallet.

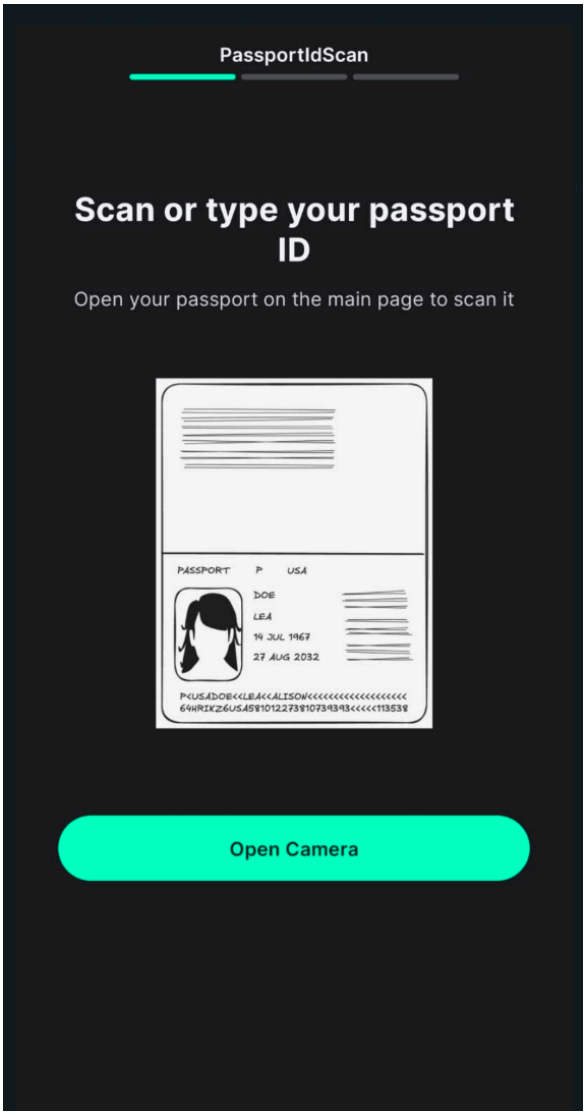
Requires a one-time blockchain transaction to store commitments.

The wallet must contain enough ETH to pay for the one-time gas fee.

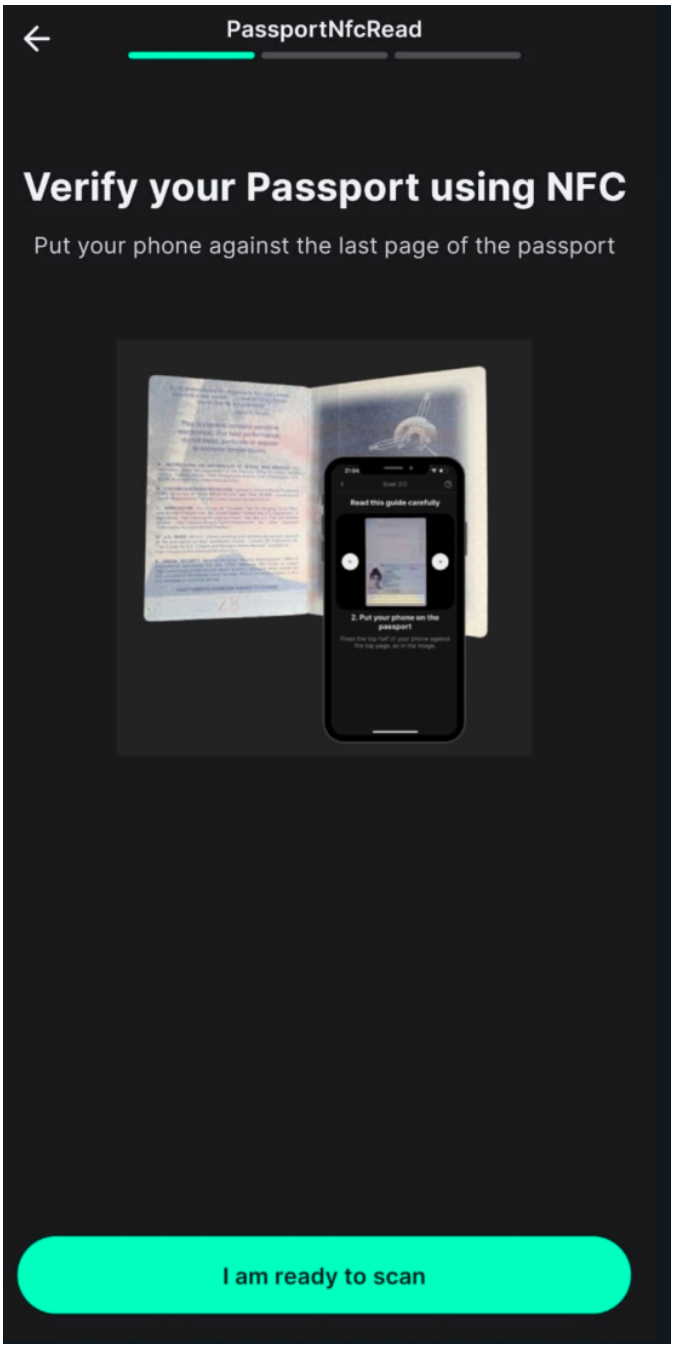
Identity Verification Workflow

In the mobile application:

1. Scan your passport number through the camera. Tap “Open Camera” and center your password for the camera.



2. Use NFC by holding your phone near your passport chip. Keep slowly moving your passport until you feel the phone's vibration, then stop moving.



After successful scanning, a cryptographic commitment is generated. You can export the commitment to your web extension wallet

On the extension: Use the commitment to verify your identity with online services.

Tips and Troubleshooting

Use proper lighting for camera-based MRZ scanning.

When scanning NFC, keep slowly moving the passport until you feel the vibration.

Wait for confirmation before moving the passport away; keep in mind it can take a couple of seconds.

Keep your wallet key and export phrases confidential.

Security Recommendations

Never share your wallet key or export phrase with anyone.

Only install APKs from trusted sources.

Always log out or close the extension when not in use.

Use strong device and application-level security.