ENHANCED QR CODE-BASED CERTIFICATE AUTHENTICATION MODEL UTILIZING ELLIPTIC CURVE CRYPTOGRAPHY (ECC), ZERO-KNOWLEDGE PROOF (ZKP), AND MULTI-FACTOR AUTHENTICATION (MFA)

Programs
College of Informatics and Computing Sciences
BATANGAS STATE UNIVERSITY
The National Engineering University
Batangas City

In Partial Fulfillment
Of the Requirements for the Degree of
Master of Science in Information Technology

By:

Jerome B. Geli

September 2025

APPROVAL SHEET

This thesis entitled **ENHANCED QR CODE-BASED CERTIFICATE AUTHENTICATION MODEL UTILIZING ELLIPTIC CURVE CRYPTOGRAPHY (ECC), ZERO-KNOWLEDGE PROOF (ZKP), AND MULTI-FACTOR AUTHENTICATION (MFA)** prepared and submitted by **JEROME B. GELI** in partial fulfillment of the requirements for the degree Master of Science in Information Technology has been examined and is recommended for acceptance and approval for Oral Examination.

MARICEL GRACE Z. FERNANDO, DIT
Adviser

Approved by the Committee on Oral Examination with a grade of __________.

PANEL OF EXAMINERS

PRINCESS MARIE B. MELO, DIT
Panel Chairperson

Panel Member

Panel Member

External Panel Member

Accepted and approved in partial fulfillment of the requirements for the degree of Master of Science in Information Technology.

____
Date

PRINCESS MARIE B. MELO, DIT
Dean, CICS

ACKNOWLEDGEMENT

I would like to express my gratitude to the following for giving me a great opportunity towards another academic milestone that I believe will establish my scholarly excellence and success upon the completion of this study:

First to Almighty GOD, who directed and strengthened me in all endeavors, the eternal love and forgiveness bestowed upon me have been the source of wisdom and motivation, essential for the realization of this. I want to express my heartfelt appreciation to the following individuals for their steadfast aid and support in my study.

To Dr. Princess Marie B. Melo, DIT, my professor in Thesis 1, for her assistance and for sharing her expertise.

To Dr. Maricel Grace Z. Fernando, my adviser for his time, effort, for her invaluable guidance, unwavering support, and scholarly insights throughout this research endeavor.

To the panelist, for their comments and recommendations to improve this project.

Most especially, to my family for their moral, financial, and spiritual support to accomplish this project.

DEDICATION

This work is dedicated with sincere gratitude to those who have been instrumental in shaping my future.

First and foremost, to the creator, ALMIGHTY GOD, the creator of all things and the essence of life and love,

To my Parents, who have given us their limitless moral and financial which has fueled my educational journey,

To my co-teachers and friends, whose invaluable assistance and shared knowledge have been instrumental in enabling me to pursue and continue this study despite their time constraints.

And to my professors for sharing their expertise, as well as the necessary competence to complete the project.

Table of Contents

CHAPTER 1

INTRODUCTION

Background of the Study

In today's rapidly evolving digital landscape, the issuance and verification of credentials play an important role in establishing both organizational and individual credibility. Credentials not only validate the successful completion of programs or qualifications but also serve as proof of a system's commitment to integrity and excellence. They provide individuals with tangible evidence of their acquired skills and knowledge, offering a competitive edge in the job market while fostering a sense of accomplishment and confidence. However, existing authentication and verification systems face significant challenges, particularly in terms of security, efficiency, and reliability.

The growing problem of credential fraud poses a significant risk to various sectors, including employment and government services. In the Philippines, cases of counterfeit credentials have been reported, where individuals obtained fake documents from unauthorized sources. For example, in 2021, the National Bureau of Investigation (NBI) uncovered a syndicate selling fake professional licenses and government certifications. These fraudulent documents were used for job applications, license renewals, and even government employment. Internationally, similar concerns have been raised. The Axact scandal in Pakistan involved a massive operation that issued thousands of fake credentials from non-existent entities, deceiving employers and institutions worldwide. In India, criminals exploited weaknesses in QR code-based verification systems by cloning QR codes that redirected verifiers to fraudulent websites, enabling unauthorized alterations of records. Such cases show how traditional verification systems can be compromised through digital forgery, social engineering, and weak authentication mechanisms.

Several studies reveal inherent flaws in existing QR code-based authentication systems, which compromise their security and effectiveness. One major concern is the lack of robust encryption, as many systems use unsecured data structures with minimal cryptographic protection. This leaves personal information vulnerable to interception and manipulation, posing risks such as data breaches, unauthorized access, and identity theft. Another issue is the weakness of digital signatures in many QR code authentication systems. Without strong cryptographic protection, digital signatures can be forged or altered, leading to fraudulent manipulation of credential data. This undermines the credibility of organizations and devalues the authenticity of verified documents. Furthermore, many existing systems rely on shared keys, which introduce security vulnerabilities. Attackers can exploit these weaknesses by replacing legitimate QR codes with counterfeit ones, directing users to fraudulent websites designed for phishing attacks. As a result, unsuspecting users may disclose sensitive personal information, compromising their privacy and security.

The ease of generating fake QR codes further worsens the problem. Malicious actors can replicate QR codes to create counterfeit credentials, leading to widespread fraud. This not only damages the reputation of organizations but also reduces trust in verification systems, with potential consequences for both individuals and institutions.

To mitigate these challenges, this study proposes an enhanced QR code-based authentication model integrating Elliptic Curve Cryptography (ECC), Multi-Factor Authentication (MFA), and Zero-Knowledge Proof (ZKP). ECC is chosen for its ability to provide high security with smaller key sizes, ensuring efficient encryption and decryption processes. A 256-bit ECC key delivers security equivalent to a 3072-bit RSA key while requiring significantly lower computational resources. This makes ECC an ideal solution for mobile and web-based authentication environments where performance and security are equally important. Additionally, ECC offers faster computation times, lower power consumption, and improved scalability, making it particularly suitable for modern applications, including IoT, blockchain, and cloud-based authentication systems. The smaller key sizes also enhance bandwidth efficiency, reducing the storage and transmission overhead required for cryptographic operations.

By leveraging ECC alongside MFA, the proposed authentication model enhances the integrity, confidentiality, and authenticity of digital certificates. The system ensures that certificates are protected from unauthorized alterations, while the use of MFA strengthens access control mechanisms, preventing unauthorized access and fraudulent verification attempts. Furthermore, the integration of Zero-Knowledge Proof (ZKP) introduces another layer of security and privacy by enabling verification without exposing sensitive certificate data. Unlike conventional authentication methods that may require revealing portions of private information during verification, ZKP allows users to prove certificate validity without disclosing any data. This significantly mitigates risks related to data leaks and unauthorized access.

In summary, this study emphasizes the development of an advanced authentication model that enhances security, efficiency, and reliability in certificate verification with its implementation demonstrated in the education sector, specifically for securing the certificate authentication process against fraudulent activities.

Objectives of the Study

The main objective of this study is to design and develop an enhanced certificate authentication model that enhances the integrity, authenticity, and protection of certificates, providing mechanisms for verifying their legitimacy and providing reliable access to certificate information.

Specific Objectives:

1. To implement a QR code-based authentication module that securely encodes certificate data using strong encryption, digital signatures, and validation mechanisms.
2. To enhance certificate authentication using Elliptic Curve Cryptography (ECC), Zero-Knowledge Proof (ZKP), and Multi-Factor Authentication

(MFA), employing cryptographic techniques for secure data protection, efficient verification, and privacy-preserving authentication.

3. To develop a certificate management system for the verification of credentials by the applicants and employers and enables officer/s to:
   3.1 oversee all system operations,
   3.2 manage certificate issuance, regulate user access, monitor system activities, and perform maintenance tasks to ensure the secure and efficient administration of records.

Significance of the study

This study highlights the need for an improved certificate authentication process to address the growing risks of credential fraud, identity theft, and unauthorized alterations. By enhancing QR code-based certificate authentication with Elliptic Curve Cryptography (ECC), Multi-Factor Authentication (MFA), and Zero-Knowledge Proof (ZKP), the study aims to strengthen security, reliability, and efficiency in verification processes. The integration of ECC ensures that certificates are resistant to tampering, offering robust protection with smaller key sizes, reducing storage and processing overhead while maintaining strong security. The incorporation of ZKP allows for secure verification without exposing sensitive data, safeguarding user privacy while maintaining the integrity of credentials. The use of MFA further reinforces authentication by preventing unauthorized access during verification.

For different institutions, this study provides an enhanced framework for to ensure certificate authenticity, to reduce the risks associated with fraudulent credentials, and to reinforce the credibility of certifications. The adoption of cryptographic methods supports compliance with modern data protection standards, addressing concerns related to data integrity and confidentiality. In colleges and universities, graduates will benefit from a secure verification process that eliminates risks associated with forged certificates, allowing them to confidently present their qualifications to employers, government agencies, and other institutions. This improved authentication mechanism enhances accessibility, enabling global recognition of verified credentials without geographical limitations.

Employers and verification agencies gain access to a streamlined, tamper-proof certificate authentication mechanism. The integration of QR codes with cryptographic verification allows real-time validation of academic credentials, reducing hiring risks associated with fraudulent qualifications. With MFA and ZKP, employers can verify certificates with confidence, improving efficiency in recruitment and background verification processes. Beyond its application in diverse certificates, the study contributes to advancements in secure digital identity verification, demonstrating the effectiveness of ECC, MFA, and ZKP in broader authentication scenarios, such as government-issued documents, professional licenses, and identity management systems.

By addressing vulnerabilities in certificate verification, this study lays the foundation for a highly secure, scalable, and privacy-preserving model that enhances trust in credentials and strengthens digital authentication frameworks. The findings will provide valuable insights into modern cryptographic authentication strategies, paving the way for more secure, efficient, and fraud-resistant credential verification systems.

Scope and Limitations

The scope of this project focuses on the digitization of certificate authentication and verification for college graduates. The proposed QR code- based authentication model enhances security and efficiency by integrating Elliptic Curve Cryptography (ECC), Multi-Factor Authentication (MFA), and Zero-Knowledge Proof (ZKP) to address vulnerabilities in QR code-based authentication systems. The system is designed to facilitate the authentication of academic degrees, diplomas, certificates, and transcripts. It provides a secure platform for institutions to issue certificates, embed QR codes for verification, and enable graduates and employers to authenticate credentials through an efficient and tamper-resistant process.

The system does not handle the verification of individuals who register on the platform to confirm their credential status. This must be conducted manually or through a separate management system. However, if the existing management system provides an API and is integrated with the authentication platform, graduates or users do not need to register again, accounts are automatically created and linked based on existing records. For employers, if they already have an account or if the organization is in partnership with the employer, there is no need to request access again. The system does not process payments for new credential requests, as financial transactions remain under the organization's online payment system. The platform operates exclusively in an internet-based environment, requiring a stable connection for optimal functionality. It does not replace official institutional policies for credential validation but serves as an additional digital verification layer. All verification-related communications, including multi-factor authentication (MFA) codes, are sent through registered email.

To uphold data privacy standards, the system incorporates encryption, strict access controls, and audit trails to ensure the confidentiality and integrity of personal and academic information. All data transmissions, including certificate uploads and QR code verifications, occur over secure communication channels to prevent interception or unauthorized access. Only authorized personnel have access to sensitive information, with authentication mechanisms in place to verify user identities. The system follows the principle of data minimization by collecting only essential information required for verification, reducing exposure to potential security risks. Audit logs are implemented to monitor system activities, reinforce accountability, and support compliance with data protection regulations. Graduates and institutions are required to provide explicit consent for data processing within the system, and regular security assessments are conducted to align with evolving data privacy laws.

By defining these parameters, the project establishes a clear roadmap for development, prioritizing security, efficiency, and accessibility. The integration of ECC, ZKP, and MFA, not only enhances authentication but also ensures that the verification process is resistant to fraud, unauthorized modifications, and data breaches, providing a reliable and scalable solution for institutions and their stakeholders.

CHAPTER II

REVIEW OF RELATED STUDIES AND SYSTEMS

This chapter discusses the facts, concepts and principles that have direct bearing or relation to the conceptualization of the design project. The related literature includes the critical view, and evaluation of related studies from readings, write-ups and previous projects. It also emphasizes the technical concepts that are essential in the development of the project.

Technical Background

This section discusses the technicalities and concepts applied in the project's system analysis and design. The concepts discussed below will serve as the framework of the project.

QR Code Generation and Parsing

QR (Quick Response) codes are two-dimensional barcodes that store data in a matrix pattern. They can encode various types of information, such as text, URLs, or other data. QR code generation involves creating these codes programmatically, often using libraries or APIs, while parsing involves extracting the encoded information from a QR code. The utilization of QR codes in authentication systems offers advantages in terms of ease of use and compatibility with mobile devices, enhancing the user experience.

Elliptic Curve Cryptography (ECC)

Elliptic Curve Cryptography (ECC) is a form of asymmetric cryptography that ensures confidentiality, integrity, and authenticity of data. It operates using elliptic curve mathematics over finite fields, allowing for stronger security with smaller key sizes compared to traditional methods. ECC is widely used in encryption, digital signatures, and authentication protocols, making it an essential component of the proposed system for secure certificate authentication and verification.

Digital Signatures

Digital signatures are cryptographic mechanisms used to verify the authenticity and integrity of digital messages or documents. They provide a way for the recipient of a message to verify that the claimed sender sent it and that it has not been altered during transit. Digital signatures are based on asymmetric cryptography, where a private key is used to sign a message, and a corresponding public key is used to verify the signature. This ensures non-repudiation and data integrity, crucial aspects in secure authentication systems.

Hashing and SHA 256

Hashing is a cryptographic process that converts data into a fixed-length string of characters, which typically appears random. Unlike encryption, hashing is a one-way function, meaning the original data cannot be retrieved from the hash value. SHA-256 (Secure Hash Algorithm 256-bit) will be employed for generating hashes of data such as certificate contents or transaction logs. SHA-256 is widely recognized for its strong collision resistance and computational security. It helps ensure data integrity by enabling the detection of any unauthorized modifications and is essential in signing and verifying digital certificates in combination with ECC.

Zero-Knowledge Proof (ZKP)

Zero-Knowledge Proofs are a cryptographic method that prove someone that they have the right information without showing what that information is. Unlike traditional methods where verification often requires partial or full disclosure, ZKPs keep sensitive information completely hidden throughout the process. This makes them ideal for systems where privacy is a priority. When applied to certificate verification, ZKPs can confirm the legitimacy of a user's credentials without exposing personal details, adding a layer of confidentiality that complements existing security features like digital signatures and encryption

Multi-Factor Authentication (MFA)

Multi-Factor Authentication (MFA) is a security process that requires users to provide two or more types of verification before gaining access to a system. These factors typically include something the user knows (like a password), something they have (like a mobile device or security token), or something they are (like a fingerprint). By combining multiple forms of authentication, MFA significantly reduces the risk of unauthorized access, even if one factor is compromised. In the proposed system, MFA strengthens user identity verification, allowing only authorized individuals to access or request credentials, which contributes to overall system security.

Secure Transmission Protocols

Secure transmission protocols, such as HTTPS (Hypertext Transfer Protocol Secure), TLS (Transport Layer Security), and SSH (Secure Shell), provide mechanisms for encrypting data during transmission over a network. These protocols establish secure communication channels between clients and servers, protecting against eavesdropping, tampering, and man-in-the-middle attacks. Implementing secure transmission protocols is essential for safeguarding sensitive data transmitted within the system.

Application Programming Interface (API)

Application Programming Interface (API) is a structured set of rules and protocols that enable different systems to communicate and share data seamlessly. APIs allow existing school management systems to integrate directly with the certificate authentication platform. In the proposed system, the API facilitates the automatic transfer of graduate information from the school’s database to the authentication system, to ensure that certificate data is consistently synchronized, accurate, and ready for secure processing. This reduces redundancy, improves efficiency, and enhances user experience by eliminating the need for graduates to register separately.

Database Management

Database management involves the organization, storage, retrieval, and manipulation of data in a structured format within a database system. It includes tasks such as database design, implementation, optimization, and maintenance to ensure efficient and reliable data management. Securely storing user credentials, access logs, and authentication metadata in a database is essential for the functioning of the system. Utilizing database management systems (DBMS) with robust security features is recommended to mitigate data breaches and unauthorized access.

User Interface Design

User interface (UI) design focuses on creating intuitive and user-friendly interfaces for interacting with software applications or systems. It involves designing layouts, navigation menus, input forms, and visual elements to enhance usability and accessibility. A well-designed user interface is important in delivering a positive user experience within the system.

Web Development

Web development encompasses the processes of designing, building, and maintaining websites or web applications. It involves various technologies, frameworks, and programming languages such as HTML, CSS, JavaScript, and server-side scripting languages (e.g., PHP, Python, Ruby). Developing a secure and responsive web interface for the system requires expertise in web development principles, security best practices, and user experience design.

System Architecture

System architecture refers to the high-level structure and design of a software system, including its components, interactions, and deployment considerations. It involves defining the system's modules, layers, interfaces, and dependencies to meet functional and non-functional requirements effectively. Designing a scalable, modular, and secure architecture for the system is essential to achieve reliability, performance, and maintainability. Utilizing architectural patterns such as MVC (Model-View-Controller) or microservices can help in organizing and managing the complexity of the system.

QR Codes in Authentication and Certificate Verification

QR codes (Quick Response codes) were first developed by Denso Wave, a subsidiary of Toyota, in 1994 for tracking automotive parts. Over time, they have gained popularity across various fields due to their capability to store and transmit a wide range of data, from URLs to personal identification information. A QR code consists of black and white squares arranged in a grid, which can store up to 4,296 alphanumeric characters, far exceeding the capacity of a traditional barcode.

In the context of authentication, QR codes are increasingly being employed as a secure and efficient way to authenticate user identities or validate documents, especially certificates. Many organizations and certification bodies use QR codes to encode information such as the certificate holder's details, the course or program completed, and the institution’s authentication keys. When scanned, these codes provide immediate access to an online database or verification service, where the details can be cross-checked.

One significant advantage of using QR codes for authentication is their ease of generation and readability using common mobile devices. The verification process typically involves scanning the QR code using a smartphone or a QR code reader, which retrieves the embedded data. This data is then matched against a secure backend database managed by the issuing organization, allowing employers or third parties to instantly verify the authenticity of a certificate.

In some advanced systems, QR codes are integrated with blockchain technology or public-key cryptography (such as Elliptic Curve Cryptography, ECC) to further enhance security. This approach ensures that once the QR code is generated, any attempt to modify the information it encodes will render it invalid. The use of QR codes for certificate verification also combats the growing problem of fake degrees and certifications, as it provides a reliable and tamper-evident method for verifying credentials.

In addition to certificates, QR codes are also utilized in two-factor and multi-factor authentication systems. For example, Google Authenticator and other mobile apps use QR codes to initiate a secure connection between the user’s device and an online service. In this scenario, the QR code serves as a secondary authentication factor, complementing traditional methods like passwords, and significantly improving overall security by reducing the risk of phishing attacks or credential theft.

ECC in Authentication and Certificate Verification

Elliptic Curve Cryptography (ECC) is a public-key cryptographic system that offers strong security with smaller key sizes, making it more efficient in terms of computation and storage. ECC is based on the mathematical properties of elliptic curves over finite fields, where its security relies on the difficulty of solving the Elliptic Curve Discrete Logarithm Problem (ECDLP). This makes ECC a preferred choice for secure communication, authentication, and certificate verification.

For certificates, ECC is used to sign sensitive certificate data, such as the user's identity, other information, and the issuing institution's signature. The institution generates a private key to sign the certificate and a corresponding public key for verification. The signed certificate is stored in a QR code or a secure database. When a third party needs to verify the certificate, they use the institution's public key to validate the digital signature.

ECC-Based Certificate Authentication Process

1. Certificate Generation
   - The institution generates a digital certificate containing the user's details (example in graduates: name, course, date of completion).
   - The certificate data is signed using the institution's private ECC key, producing a digital signature.

    - The signed certificate is converted into a QR code or stored in a secure database for retrieval.

2. Certificate Verification
    - A third party (e.g., an employer or another institution) scans the QR code or accesses the certificate online.
    - The system retrieves the digital signature and verifies it using the institution’s public ECC key.
    - If the verification is successful, it confirms that the certificate is authentic and has not been tampered with.
3. Tamper Resistance
    - ECC provides strong tamper resistance due to its cryptographic properties.
    - Any modification to the certificate after signing will cause the verification process to fail.

Beyond certificate verification, ECC is widely used in secure authentication mechanisms, including secure login systems and encrypted communications. It is integral in Secure Socket Layer (SSL) and Transport Layer Security (TLS) protocols, safeguarding data transmission. Due to its efficiency, smaller key sizes, and strong security, ECC is an ideal choice for authentication and certificate verification, providing a scalable and secure solution for digital credentialing systems.

Zero Knowledge Proof (ZKP) in Authentication and Certificate Verification

Zero-Knowledge Proof (ZKP) is a cryptographic protocol that enables one party (the prover) to convince another party (the verifier) that a given statement is true without disclosing any sensitive information. This concept is particularly useful in certificate authentication and verification, where the goal is to confirm a certificate's legitimacy without exposing private keys, certificate details, or cryptographic signatures.

In ZKP-based authentication, when a certificate is issued, proof is generated based on a secret mathematical relationship tied to the certificate data. The verifier, upon scanning a QR code or accessing the certificate, checks the proof against a predefined validation function. If the proof is valid, the system confirms authenticity without revealing the original certificate details.

Key advantages of using ZKP in certificate authentication include:

- Confidentiality - The actual certificate data and cryptographic keys remain hidden from verifiers.
- Forgery Resistance - The proof is mathematically linked to the certificate but cannot be forged without access to the legitimate certificate's confidential information.
- Efficiency - Instead of decrypting signatures, the verifier checks a cryptographic proof, making verification faster and computationally efficient.

- Data Integrity - Any modification to the certificate invalidates the proof, meaning only unaltered certificates are accepted.

Technical Comparison: ECC-based only vs. ZKP-based Verification

The following table outlines the technical differences between traditional ECC-based and ZKP-based certificate verification:

Table 1. *Technical Comparison of ECC only vs. ZKP-Based Verification*

Feature	**Traditional ECC-based Certificate Verification**	**ZKP-Based Certificate Verification**
Verification Method	Digital signature verification using ECC	Cryptographic proof verification using ZKP
Data Exposure	Public keys and certificate data are exposed	Only proof and unique identifier are revealed
Encryption/Decryption	Requires decryption using a public key	No decryption needed, verification via proof
Forgery Resistance	Can be faked if private key leaks	Nearly impossible to forge without original data
Storage Requirements	Stores full certificate and digital signature	Stores only unique identifier and ZKP proof
Verification Complexity	Computationally expensive	More lightweight, quick verification
Mathematical Basis	ECC (Elliptic Curve Cryptography)	Interactive or non-interactive ZKP schemes (e.g., zk-SNARKs, zk-STARKs, Schnorr ZKP)
Scalability	Higher overhead in large-scale verification	Optimized for scalability in decentralized and secure systems

ECDSA: Elliptic Curve Signatures

The ECDSA (Elliptic Curve Digital Signature Algorithm) is a cryptographically secure digital signature scheme, based on the elliptic-curve cryptography (ECC). ECDSA relies on the math of the cyclic groups of elliptic curves over finite fields and on the difficulty of the ECDLP problem (elliptic-curve discrete logarithm problem). The ECDSA sign / verify algorithm relies on EC point multiplication and works as described below. ECDSA keys and signatures are shorter than in RSA for the same security level. A 256-bit ECDSA signature has the same security strength as 3072-bit RSA signature.

ECDSA uses cryptographic elliptic curves (EC) over finite fields. These curves are described by their EC domain parameters, specified by various cryptographic standards such as SECG: SEC 2 and Brainpool (RFC 5639). Elliptic curves, used in cryptography, define:

- Generator point G, used for scalar multiplication on the curve (multiply integer by EC point)
- Order *n* of the subgroup of EC points, generated by G, which defines the length of the private keys (e.g. 256 bits)

  For example, the 256-bit elliptic curve secp256k1 has:

  Order *n* =
  115792089237316195423570985008687907852837564279074904382605
  163141518161494337 (prime number)

  Generator point G {*x* =
  550662630222773436695787188951685343262506034537775941755001
  87360389116729240, *y* =
  326705100207588169780830851305070431844712733806592432759389
  04335757337482424}

  Key Generation

  The ECDSA key-pair consists of:

  - private key (integer): *privKey*
  - public key (EC point): *pubKey* = *privKey* * G

The private key is generated as a random integer in the range [0...*n*-1]. The public key *pubKey* is a point on the elliptic curve, calculated by the EC point multiplication: *pubKey* = *privKey* * G (the private key, multiplied by the generator point G). The public key EC point {*x*, *y*} can be compressed to just one of the coordinates + 1 bit (parity). For the secp256k1 curve, the private key is 256-bit integer (32 bytes) and the compressed public key is 257-bit integer (~ 33 bytes).

ECDSA Sign

The ECDSA signing algorithm (RFC 6979) takes as input a message *msg* ****+ a private key *privKey* ****and produces as output a signature, which consists of pair of integers {*r*, *s*}. The ECDSA signing algorithm is based on the ElGamal signature scheme and works as follows (with minor simplifications):

1. Calculate the message hash, using a cryptographic hash function like SHA-256: *h* = hash(*msg*)
2. Generate securely a random number *k* in the range [1..*n*-1]
    - In case of deterministic-ECDSA, the value *k* is HMAC-derived from *h* + *privKey* (see RFC 6979)
3. Calculate the random point *R* = *k* * G and take its x-coordinate: *r* = *R*.x

4. Calculate the signature proof: *s* =

   k−1∗(h+r∗privKey)(modn)k−1∗(h+r∗privKey)(modn)

   - The modular inverse k−1(modn)k−1(modn) is an integer, such that k∗k−1≡1(modn)k∗k−1≡1(modn)

5. Return the signature {*r*, *s*}.

The calculated signature {*r*, *s*} is a pair of integers, each in the range [1...*n*-1]. It encodes the random point *R* = *k* * G, along with a proof *s*, confirming that the signer knows the message *h* and the private key *privKey*. The proof *s* is by idea verifiable using the corresponding *pubKey*. ECDSA signatures are 2 times longer than the signer's private key for the curve used during the signing process. For example, for 256-bit elliptic curves (like secp256k1) the ECDSA signature is 512 bits (64 bytes) and for 521-bit curves (like secp521r1) the signature is 1042 bits.

ECDSA Verify Signature

The algorithm to verify a ECDSA signature takes as input the signed message *msg* + the signature {*r*, *s*} produced from the signing algorithm + the public key *pubKey*, corresponding to the signer's private key. The output is boolean value: *valid* or *invalid* signature. The ECDSA signature verify algorithm works as follows (with minor simplifications):

1. Calculate the message hash, with the same cryptographic hash function used during the signing: *h* = hash(*msg*)

2. Calculate the modular inverse of the signature proof: *s1* = s−1(modn)s−1(modn)
3. Recover the random point used during the signing: *R′* = (*h* * s1) * G + (*r* * *s1*) * *pubKey*
4. Take from *R′* its x-coordinate: *r′* = *R′*.x
5. Calculate the signature validation result by comparing whether *r′* == *r*

The general idea of the signature verification is to recover the point *R′* using the public key and check whether it is same point *R*, generated randomly during the signing process. The ECDSA signature {*r*, *s*} has the following simple explanation:

The signing signing encodes a random point *R* (represented by its x-coordinate only) through elliptic-curve transformations using the private key *privKey* and the message hash *h* into a number *s*, which is the proof that the message signer knows the private key *privKey*. The signature {*r, s*} cannot reveal the private key due to the difficulty of the ECDLP problem.

The signature verification decodes the proof number *s* from the signature back to its original point *R*, using the public key *pubKey* and the message hash *h* and compares the x-coordinate of the recovered *R* with the *r* value from the signature.

The diagram below illustrates the ECDSA digital signature verification process:

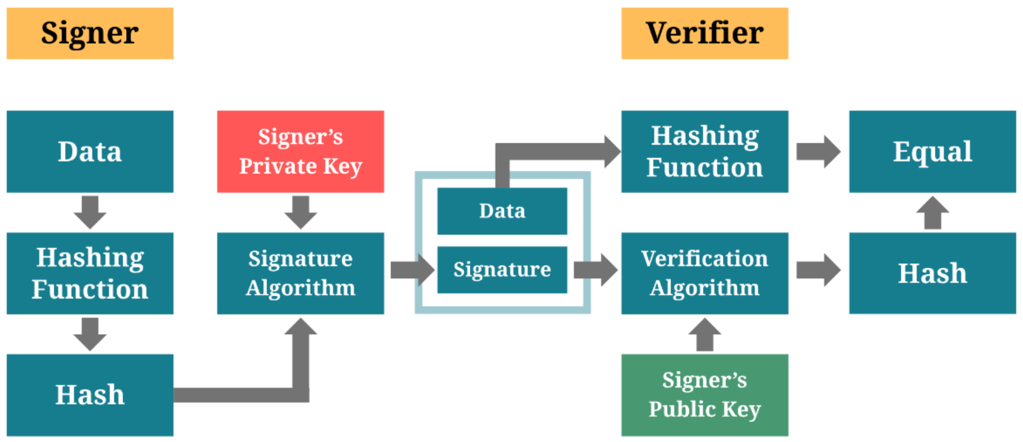


Figure 1. *ECDSA Digital Signature Verification Process*

Advantages of ECC

- Smaller Key Sizes: ECC provides the same level of security as traditional algorithms but with significantly smaller key sizes (e.g., a 256-bit ECC key is comparable to a 3072-bit key in other cryptosystems).
- Higher Efficiency: Faster encryption and decryption process due to reduced computational overhead.
- Stronger Security: The difficulty of solving ECDLP makes ECC more resistant to brute-force attacks.

With its efficiency, scalability, and strong security, ECC is a preferred cryptographic method for authentication and certificate verification, providing a reliable and tamper-resistant solution for modern digital security challenges.

Comparison of ECC and RSA Encryption

Elliptic Curve Cryptography (ECC) and Rivest–Shamir–Adleman (RSA) are widely used algorithm in SSL/TLS certificates. While both provide secure encryption, ECC offers stronger security with shorter key lengths, reducing computational load and improving efficiency.

Table 2. *RSA vs ECC: Key Length Comparison*

Security (In bits)	RSA Key Length Required	ECC Key Length Required
80	1024	160-223
112	2048	224-255
128	3072	256-383
192	7680	384-511
256	15360	512+

RSA, based on prime factorization, has been the standard for decades but requires significantly larger key sizes to maintain security. As computational power increases, RSA is becoming more vulnerable. ECC, which relies on elliptic curve mathematics, offers superior security with lower processing requirements, making it ideal for modern applications.

Table 3. *RSA vs ECC: Security and Performance Comparison*

Factor	RSA	ECC
Security Strength	Less secure at equivalent key sizes.	Stronger security with shorter keys.
Key Size	Requires larger keys (2048-bit+).	Smaller keys (256-bit = 3072-bit RSA).
Speed	Slower due to larger keys.	Faster due to efficient computation.
Scalability	Not ideal for low-power devices.	Optimized for mobile
Quantum Resistance	Harder to scale with increasing key sizes.	More scalable and efficient.
Adoption	Still widely used but declining.	Growing adoption in modern security.

Due to its efficiency and resistance to brute-force attacks, ECC is increasingly replacing RSA, especially in environments where performance and security are critical. Experts predict that RSA may become obsolete by 2030, making ECC a more future-proof choice for encryption in SSL/TLS certificates.

ECC Key Length and Security Efficiency

Elliptic Curve Cryptography (ECC) is a modern and highly efficient method for securing data, utilizing a pair of keys for encryption and decryption. The security of ECC depends on key length, but unlike traditional encryption methods, ECC achieves the same level of security with significantly smaller keys. Two common ECC key sizes are 256-bit and 384-bit:

- 256-bit ECC provides a level of security comparable to 3072-bit encryption.
- 384-bit ECC offers even stronger security, making it highly resistant to cryptographic attacks.

In this study, 256-bit ECC was chosen to protect digital certificates, providing strong defense against forgery, tampering, and unauthorized access. Due to ECC’s efficiency, this key size provides high security while maintaining fast processing speeds, making it an ideal choice for modern verification systems.

One major advantage of ECC is its future-proofing capability. As computational power increases and quantum computing continues to develop, traditional encryption methods may become vulnerable. ECC, with its strong security and smaller key requirements, remains a practical and scalable solution.

While quantum-resistant cryptographic methods are still being researched, adopting 256-bit ECC helps safeguard against emerging threats.

The security of certificates is important for maintaining trust among different institutions and organizations. This choice of using ECC enhances the credibility and reliability of the certificate verification process, reinforcing the integrity of the issuing institutions and the authenticity of their credentials.

QR Code and ECC Integration

The integration of QR Code technology and Elliptic Curve Cryptography (ECC) has emerged as a robust approach for enhancing the security and efficiency of digital credential verification. QR codes serve as a compact and accessible medium for encoding cryptographically protected data, while ECC provides a lightweight and highly secure method for digital signatures and encryption. When used together, they enable secure, tamper-resistant certificate authentication systems suitable for modern verification needs.

ECC is well-suited for embedding secure data within the limited storage capacity of QR codes. This makes it a preferred solution in contexts where mobile devices or resource-constrained environments are used for scanning and verification. ECC-based digital signatures (such as ECDSA) can be embedded within QR codes alongside metadata like certificate identifiers, timestamps, and issuing authority details.

Research and case studies in digital identity, education, and e-governance highlight the effectiveness of this integration. For instance, academic institutions and professional certification bodies have begun issuing digitally signed certificates with embedded QR codes. These allow third-party verifiers, such as employers or licensing agencies, to scan the code and validate the certificate's authenticity using public-key cryptography without relying on manual background checks. Since QR codes can store ECC-based signatures and hashes directly, verifiers can confirm authenticity using mobile apps or web tools even with limited internet access. This property is especially valuable in remote, rural, or under-connected areas.

The combination of QR codes and ECC addresses key concerns in digital certificate management: authenticity, integrity, ease of access, and resistance to forgery. It forms the basis for emerging frameworks in secure credential issuance, particularly in education, healthcare, and professional licensing domains.

Comparative Analysis of Existing and Proposed Systems

In the Philippines, several systems and products utilize QR codes and cryptographic methods for certificate and identity authentication. Below are examples of existing systems, followed by a comparative analysis highlighting their features relative to the proposed system.

PhilSys Check

The Philippine Statistics Authority (PSA) developed PhilSys Check, a web-based authentication platform for the national digital ID, known as PhilID. It allows users to verify identities through QR codes embedded in the PhilID, which can be scanned using a smartphone or computer. The system employs public-private key cryptography to ensure the authenticity of the ID without requiring internet connectivity during verification.

SECPA e-Verification Mobile App

The PSA also introduced the SECPA e-Verification Mobile App, designed to authenticate civil registry documents issued by the authority. This stand-alone system operates offline on Android devices, enabling users to verify the authenticity of documents by scanning QR codes that contain encrypted information.

SEEK Pass Digital ID

SEEK Pass offers a digital identity verification service integrated with JobStreet profiles. Users can verify their identity against government-issued documents, and a SEEK Pass-verified Digital ID badge is displayed on their profiles. This system enhances employer confidence in the authenticity of a candidate's identity.

National ID eVerify

The National ID eVerify platform provides quick, secure, and reliable identity verification to prevent identity theft and fraudulent activities. It streamlines the verification process for various services by utilizing the national ID system.

Table 4. *Comparison of Existing and Proposed System*

System / Feature	**QR Code Security**	**Forgery Prevention**	**Encryption Algorithm**	**Multi-Factor Authentication (MFA)**	**Zero-Knowledge Proof (ZKP)**	**Computational Efficiency**
PhilSys Check	Encrypted QR codes with public-private key cryptography	High, due to cryptographic verification	Public-private key cryptography	Not implemented	Not used	Moderate
SECPA e-Verification	Encrypted QR codes, offline verification	High, offline verification reduces tampering risks	Not specified	Not implemented	Not used	Moderate
SEEK Pass Digital ID	Verification against government-issued documents	Relies on initial document verification	Not specified	Not specified	Not used	Not specified
National ID eVerify (PhilSys)	Utilizes national ID system for verification	High, centralized national ID database	Not specified	Not specified	Not used	Not specified
TESDA Certificate Verification	QR code redirects to TESDA's official verification website showing training, assessment, renewal data	Based on central database records; relies on accurate certificate data	Not specified	Not used	Not used	High (automated, QR redirection + web-based validation)
Proposed System	ECC-signed QR codes with embedded digital signatures and certificate data	Enhanced with ECC and ZKP, reducing forgery risks	ECC (smaller key sizes, lower computation)	Integrated for additional security	Utilized for authentication without exposing private keys	High, due to ECC's efficient cryptographic operations

Related Studies and Systems

The increasing reliance on QR code-based authentication systems and digital credentials has introduced both opportunities and challenges in security. Several studies have investigated the vulnerabilities of existing systems and proposed cryptographic solutions to address them. Previous research on QR code authentication, cryptographic approaches, and real-world applications of multi-factor authentication (MFA) has significantly contributed to enhancing security measures.

QR code scams have become a serious threat to the integrity of many institutions and the safety of personal data. One major incident was the rise of "quishing" scams in the Philippines, where cybercriminals used QR codes to redirect users to malicious websites. This method, combining QR codes and phishing, has been increasingly utilized to steal login credentials, download harmful content, or install malware on victims' devices. In 2022, a scam involving fake QR codes in the admission process of a university in Manila tricked applicants into providing sensitive personal and financial information, leading to monetary losses. These incident underscores vulnerabilities in QR code technology when not properly secured. Experts recommend implementing cryptographic techniques such as digital signatures and robust encryption mechanisms to secure QR codes, in addition to conducting regular security audits and user education programs.

Credential forgery is another prevalent issue, as evidenced by the case of Noriza Dancel De Luna, a Filipina who presented falsified academic credentials in her application for permanent residency in Singapore in 2020. Verification by the Singapore Immigration and Checkpoints Authority (ICA) revealed that the diploma and transcript she submitted were counterfeit. Such incidents emphasize the weaknesses in traditional document verification and the need for advanced authentication measures. Research by Johnson (2019) further supports this, indicating that counterfeit academic documents are increasing, particularly where verification systems are weak. In response, different organizations are now exploring cryptographic security models, including Elliptic Curve Cryptography (ECC) and Multi-Factor Authentication (MFA), to enhance credential verification.

MFA has been widely adopted across various sectors to reinforce authentication mechanisms against security threats. In government ID verification, MFA is used in national ID systems, e-passports, and tax portals. Estonia's digital ID system integrates biometric authentication with cryptographic smart cards for secure identity validation. Similarly, tax agencies such as the Bureau of Internal Revenue (BIR) in the Philippines and the IRS (USA) mandate OTP-based authentication to prevent identity fraud. In financial transactions, MFA ensures the security of online banking and digital payments. Banks require users to verify transactions via OTPs, biometric scans, or cryptographic authentication tokens such as YubiKey. Cryptocurrency exchanges like Coins.ph and Binance also implement MFA to safeguard user accounts. In enterprise security, organizations enforce MFA for access to cloud-based applications and corporate networks, mitigating unauthorized entry risks. Tech giants like Google and Microsoft require MFA through hardware-backed security keys and authentication apps. In healthcare, MFA secures electronic health records (EHRs), telemedicine platforms, and medical devices, allowing only authorized personnel to retrieve sensitive data.

Despite advancements in QR code authentication, several vulnerabilities persist. QR code tampering and spoofing remain significant threats, as attackers can replace legitimate QR codes with fraudulent ones to redirect users to phishing sites.

Studies indicate that cybercriminals frequently use "quishing" tactics, embedding malicious URLs into QR codes to execute malware downloads or credential theft. Dependency on internet connectivity is another limitation, as many QR-based verification systems rely on online servers for validation, making them vulnerable to network failures. Offline authentication solutions, such as ECC-signed QR codes, help mitigate this issue. Encryption and data security concerns also persist, with some QR codes storing plaintext or weakly encrypted information, making them susceptible to data interception. Researchers recommend the use of digitally signed QR codes to ensure cryptographic integrity, embedding digital signatures to verify authenticity and prevent forgery. Social engineering risks pose another challenge, as many users scan QR codes without verifying legitimacy, increasing susceptibility to phishing attacks. Security education, browser-based scanning alerts, and cryptographic verification can help mitigate these risks.

Several studies propose solutions to enhance QR code authentication security. The study "A Secure QR Code System for Sharing Personal Confidential Information" by Ahamed & Mustafa (2019) presents a cryptographic model leveraging RSA for validation and verification. The authors suggest encrypting QR code data using public key cryptography to ensure confidentiality and authenticity. However, RSA has performance limitations due to its large key sizes, which is why recent studies favor ECC. "An Improved Secure QR Code Authentication System Using ECC" (Nguyen et al., 2021) demonstrates that ECC-based QR authentication reduces computational overhead while maintaining strong security. The study concludes that ECC is more efficient for mobile applications, supporting the choice of ECC over RSA.

Another relevant study, "Enhancing Certificate Authentication with ECC and Multi-Factor Authentication" (Patel & Sharma, 2022), introduces a hybrid approach combining ECC with TOTP-based MFA. Their findings suggest that integrating ECC-signed QR codes with MFA significantly improves authentication resilience. Similarly, "Lightweight Cryptographic QR Code Authentication Model Using ECC" (Chowdhury et al., 2020) highlights the benefits of encrypting certificate data within QR codes using ECC private keys. A public-private key verification model is applied to reinforce this approach to secure credential validation.

Zero-Knowledge Proof (ZKP) is another cryptographic mechanism gaining traction in authentication systems. ZKP allows one party to prove knowledge of a secret without revealing the actual information. A study by Ben-Sasson et al. (2018), "Scalable Zero-Knowledge Proofs for Blockchain Applications," explores how ZKP can enhance privacy and security in digital verification systems. Similarly, "ZKBoo: Practical Zero-Knowledge Proofs for Lightweight Devices" (Giacomelli et al., 2016) presents a practical implementation of ZKP for authentication in constrained environments, which aligns with the need for efficient mobile-based authentication. Another study, "Application of Zero-Knowledge Proofs in Secure Authentication Systems" (Almeida et al., 2021), demonstrates how ZKP can be integrated with existing cryptographic techniques such as ECC to enable identity verification without exposing private keys. These studies reinforce the applicability of ZKP in enhancing authentication within digital credentialing systems.

Various industries have implemented QR code verification systems. For example, the pharmaceutical industry utilizes QR codes to verify the authenticity of medications, helping consumers receive genuine products. The automotive industry applies QR codes to track vehicle components and prevent counterfeit parts from entering supply chains. In the Philippines, banks and fintech companies such as GCash and PayMaya have implemented QR code-based financial transactions with added authentication layers.

The implementation of these cryptographic advancements aligns with current research on secure authentication models. By signing QR codes with ECC digital signatures, unauthorized modifications are prevented. During verification, users must also complete an email-based OTP challenge as an additional MFA layer. If a QR code is altered, the ECC signature will no longer match, making the validation tamper-proof. Unlike conventional QR-based verification that depends on real-time connectivity, offline validation through cryptographically signed QR codes increases accessibility and reliability. The adoption of ECC, MFA, and Zero-Knowledge Proof in QR code-based authentication systems enhances security, mitigates risks associated with RSA-based models, and prevents credential forgery. By integrating these cryptographic techniques, a highly secure, scalable, and tamper-resistant authentication framework is achieved, addressing the major limitations of conventional QR-based verification models.

Conceptual Framework

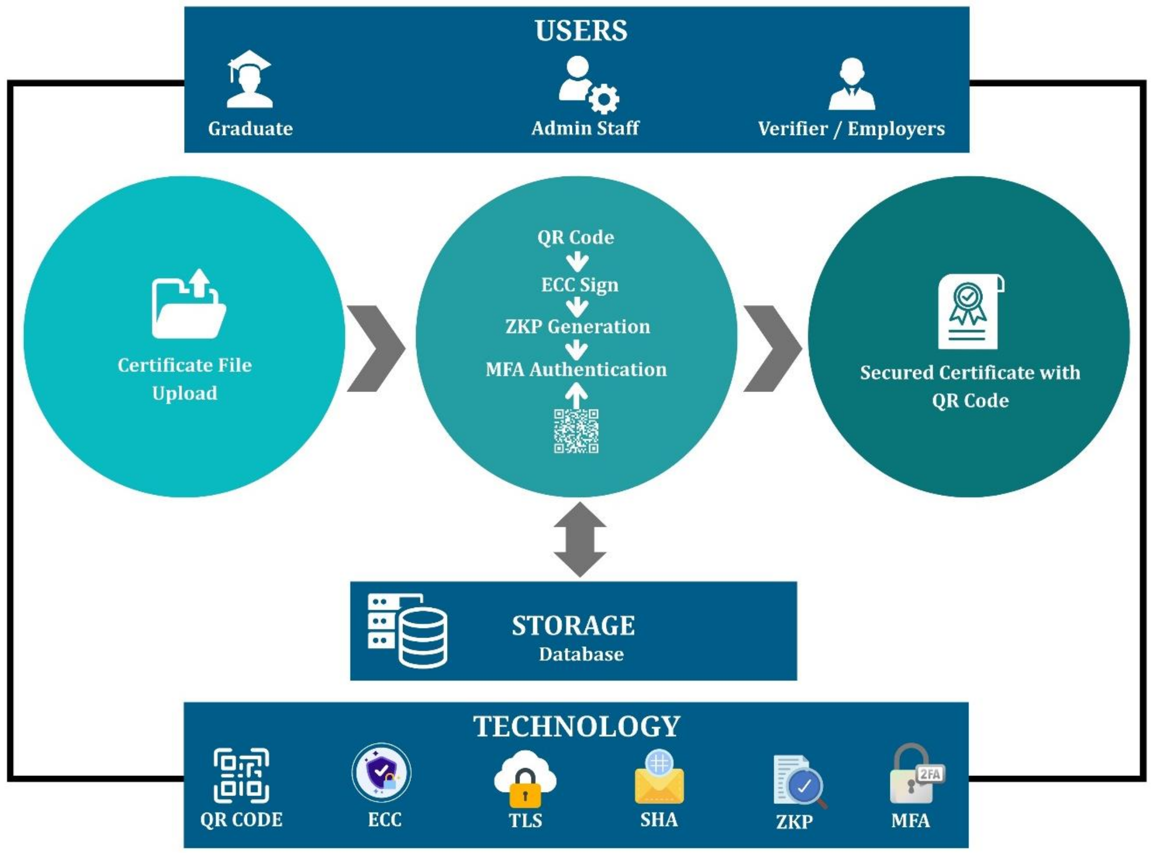


Figure 2. *Conceptual Framework*

The proposed system, Enhanced Certificate Authentication Model Utilizing QR Codes, ECC, ZKP, and MFA, is built to address modern challenges in certificate fraud, verification inefficiency, and data security. The conceptual framework captures how the different users, modules, and technologies interact to achieve secure and verifiable certificate issuance and validation. It highlights the flow of data, security layers, and verification mechanisms embedded in the system.

User Roles and Access Points

The system supports three main types of users, each interacting with the system differently:

- Students/Graduates

  Request certificates and access their authenticated versions embedded with QR codes. They act as certificate holders and sharers.

- School Administrators (Registrar, Staff)

  Responsible for uploading certificate information, digitally signing it with the institution's ECC private key, and issuing certificates through the backend interface.

- Employers/Verifiers (e.g., HR Officers)

  Scan the QR code on a certificate and verify its authenticity through a secured web interface with added Multi-Factor Authentication.

Certificate Creation, Signing, and Commitment Generation

Upon a graduate's request, the administrator performs the following:

- Certificate Upload: Certificate data (e.g., name, degree, graduation date) is submitted via the GUI.
- SHA-256 Hashing: A cryptographic hash is generated.
- ECC Digital Signing: The hash is signed using Elliptic Curve Digital Signature Algorithm (ECDSA) and the institution's private key, confirming that the certificate originated from a trusted source.

- ZKP Generation: A Zero-Knowledge Proof is computed based on the hashed data. This allows later verification without revealing sensitive content or the private key.
- Data Packaging: The system then combines the hash, ECC signature, and ZKP into a verifiable payload for encoding.

QR Code Generation and Embedding

- The combined certificate payload is converted into a byte stream, then augmented with header information and error-correction bytes.
- A masking algorithm is applied to optimize QR readability.
- Using a high error-correction level, a QR code is generated.
- This QR code is embedded into the final certificate (PDF/image), allowing offline and online verification through scanning.

Verification Process with ZKP and ECC

When an employer or verifier receives a certificate:

- They scan the embedded QR code using the verifier web app.
- The system extracts the encoded data, performs error correction, and separates the digital signature, hash, and ZKP.
- Using the institution's ECC public key, the system verifies:
  - That the digital signature is valid.
  - That the recomputed hash matches the one in the signature.
  - That the ZKP commitment confirms the authenticity without exposing the full certificate data.

- If all checks pass, the certificate is flagged as authentic and untampered.

Multi-Factor Authentication (MFA)

To prevent unauthorized access to certificate verification results:

- The verifier must enter a Time-Based One-Time Password (TOTP) sent via email.
- The system verifies this OTP before showing certificate validation details.
- This second authentication layer ensures that only intended verifiers can complete the verification process, even if a QR code is publicly shared.

Secure Data Transmission with TLS

Every interaction in the system is transmitted over TLS-encrypted HTTPS channels. This includes:

- Certificate uploads by administrators
- QR code scan requests
- Login and OTP submissions by verifiers

TLS ensures:

- Confidentiality: Data cannot be read by third parties during transmission.
- Integrity: Data is protected from tampering.
- Authentication: Communication only occurs between legitimate users and the verified server.

This conceptual framework integrates user roles, cryptographic mechanisms (ECC, ZKP), secure access (MFA), and encrypted transmission (TLS) into a unified system. Each component plays a critical role in the trust model to ensure that certificates are not only valid but tamper-proof, verifiable, and protected throughout their lifecycle. This framework provides the conceptual foundation for building a robust certificate verification system that addresses current challenges in fraud detection, manual validation, and digital security.

CHAPTER III

DESIGN AND METHODOLOGY

This chapter covers the proposed system's design and methodology, which includes the following sections: Requirement Analysis, Software Requirement Specification, and System Analysis, and Design.

Software Methodology Model

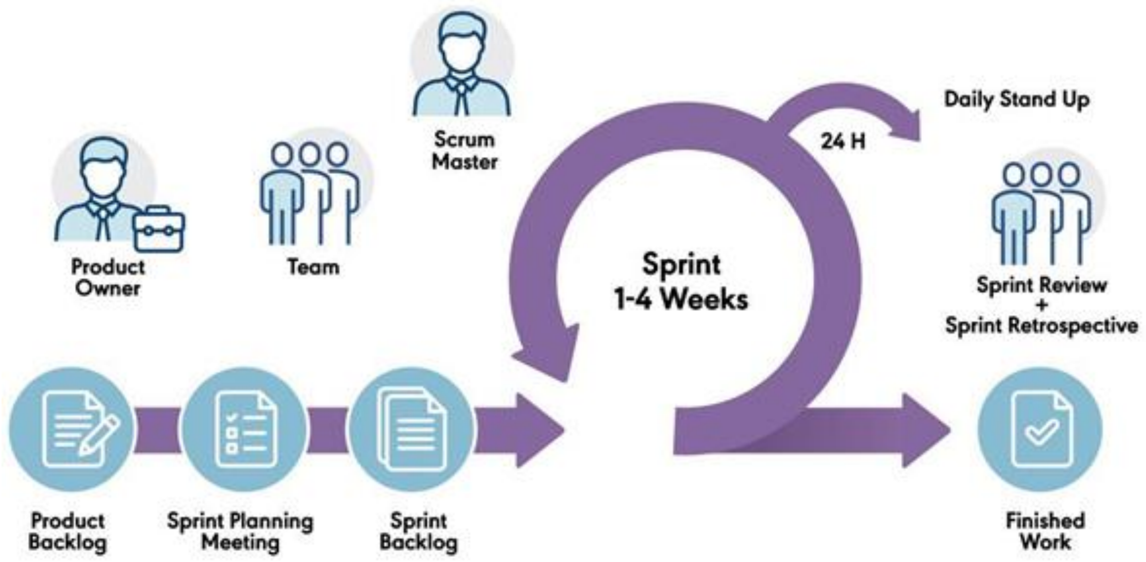


Figure 3. *Agile Scrum Methodology*

The development of this certificate authentication system adopts the Agile Scrum methodology, a structured iterative approach that emphasizes incremental delivery, rapid feedback, and continuous improvement. Scrum provides clearly defined roles, product owner (system initiator), scrum master (project lead), and development team (programmers), to facilitate collaboration.

The project is divided into sprints, each lasting two weeks, focusing on specific functional modules such as QR code generation, ECC integration, ZKP proof validation, and MFA setup. A product backlog is maintained and refined throughout the development to reflect user feedback and evolving security requirements.

At the start of each sprint, sprint planning meetings are held to define goals and assign tasks. During the sprint, daily standups help monitor progress and resolve blockers. At the end of the sprint, a sprint review is conducted with institutional stakeholders to demonstrate completed features and gather feedback. A sprint retrospective follows to evaluate team performance and identify areas for improvement.

This development approach supports early testing, frequent stakeholder engagement, and secure modular delivery, which keeps the system responsive to the evolving needs of institutions and employers.

Table 5. *Sprint Planning*

Sprint	Sprint Objectives	Task / Deliverables	Participants	Days Allotted	Status
1	Requirement Analysis	Stakeholder meeting - Define use cases and roles - Identify QR, ECC, MFA, ZKP modules - Draft specs	Developers, Institution Admin, Registrar	5 days	Done
2	System Design (Architecture & Database)	- Design client-server model - Define user flows - Database schema design - Role-based access planning	Developer, Project Manager	5 days	Done
3	ECC Digital Signature Integration	- ECC key pair generation - Certificate signing - Verify ECC signatures	Developer, Project Manager	6 days	Done

Table 5 cont'd. *Sprint Planning*

4	QR Code Structure & Generator	- Define QR format - Generate signed data QR - Test QR scanning and parsing	Developer, Project Manager	5 days	Done
5	ZKP Design & Integration	- Define ZKP use case - Implement ZKP verification in certificate module	Developer, Project Manager	6 days	Done
6	MFA Design & Implementation	-Email-based MFA setup - OTP generation and validation - Secure login logic	Developer, Project Manager	5 days	Done
7	Interface & UX Design	- Design dashboards (admin, student, registrar) - Develop UI wireframes - Review with stakeholders	Developer, Project Manager	4 days	Done
8	Module Development (Part 1)	- Certificate issuance & ECC signing module - Student application handling	Developer, Project Manager	5 days	Done
9	Module Development (Part 2)	-Verifier module - ZKP integration for verification	Developer, Project Manager	5 days	Done
10	MFA & Login Finalization	-Finalize MFA login logic - Integrate into user login system	Developer, Project Manager	4 days	Done
11	Initial System Integration	-Integrate modules (QR, ECC, MFA, ZKP) -Backend /frontend linking	Developer, Project Manager	6 days	Done
12	Unit & Interface Testing	- Test each module individually - Test UI for usability	Developer, Project Manager	5 days	Done
13	System Testing & Bug Fixes	- Full system test - Integration errors & bug fixes	Developer, Project Manager	5 days	In Progress
14	User Acceptance Testing	- User demo - Feedback collection - Final refinements	Students, Admin, Registrar, Verifiers	5 days	Pending
15	Deployment & Documentation	-Deploy system - Create manuals & technical docs	Developer, Project Manager	4 days	Pending

Constraints

Cryptographic Library and Device Compatibility (ECC-specific). The implementation of ECC (Elliptic Curve Cryptography) requires modern cryptographic libraries and compatible devices. Some older browsers or mobile phones may not fully support ECC-based digital signature verification, potentially limiting accessibility for certain users.

Multi-Factor Authentication (MFA) Delivery Constraints. Due to the absence of SMS-based OTP gateways and reliance on email-based TOTP (Time-based One-Time Password), the system's MFA implementation is dependent on users having timely access to their email. This introduces a constraint for users in areas with intermittent internet connectivity or who lack email literacy.

Security Key Storage Management. The private keys used for digital signatures are stored securely in a database, but due to budget limitations, a dedicated HSM (Hardware Security Module) or advanced key management solution was not used. This is a constraint in terms of industry-grade security assurance.

Trade-offs

Decentralized Control vs. Cross-Institution Standardization: Allowing each school to manage its own certificate system gives them control and flexibility. However, this sacrifices standardization, making inter-school verification and data consistency more difficult.

Security vs. QR Code Content Length: Encoding the full certificate content into a QR code can compromise readability and scanning performance. The system prioritizes data security and practicality by encoding only a unique reference ID or encrypted hash, which can be decrypted server-side. This reduces risk but makes the system dependent on online verification.

Security vs. Performance (ECC Operations): Elliptic Curve Cryptography (ECC) provides stronger encryption with smaller key sizes, improving security. However, the cryptographic operations, especially signature generation and verification, can introduce latency in lower-end devices or during high traffic periods.

Multi-Factor Authentication vs. User Convenience: Implementing Multi-Factor Authentication (MFA) adds a robust layer of security, especially during certificate verification. However, it may inconvenience users, particularly those in areas with limited internet access or email service reliability.

Offline Accessibility vs. Real-Time Verification: Offline verification (e.g., via embedded QR content) is not feasible due to encryption and dynamic verification needs. Online verification ensures accuracy but sacrifices accessibility in low-connectivity environments.

Hardware Requirement

The development of Certificate Authentication System necessitates careful consideration of specific devices and their specifications to ensure optimal performance and reliability. The following outlines the key hardware components and their specifications essential for the successful implementation of the system:

Server Hardware. High-performance servers are essential for smooth system operation, handling concurrent user requests, and managing the centralized database efficiently. Specifications should include multi-core processors, ample RAM, and sufficient storage capacity. Recommended Specifications: Dell PowerEdge R740 with dual Intel Xeon Gold processors, 64GB RAM, and multiple SSD drives, or other servers with equivalent performance and reliability.

Database Server. A robust database server for securely storing and managing certificate details must be installed. It should offer reliability, scalability, and advanced security features. Recommended Specifications: Oracle Database Appliance X8-2S with Oracle Database Enterprise Edition, solid-state storage, and built-in data encryption, or other database servers with similar capacity and security features.

Networking Equipment. Switches, routers, and firewalls are important for network infrastructure, connectivity, and data security. Equipment should support VLANs, Quality of Service (QoS), and intrusion prevention. Recommended Specifications: Cisco Catalyst 9000 Series switches, Cisco ASA Firepower Next-Generation Firewall, or equivalent networking equipment with similar capabilities.

QR Code Generation Hardware. Devices for printing and generating QR codes should support high-resolution printing, barcode encoding, and various connectivity options. Recommended Specifications: Epson ColorWorks C3500 color label printer, or other high-resolution QR code printing devices with similar performance.

Workstations. Desktop computers or laptops are used by system administrators and staff to access and manage the system. Workstations should have sufficient processing power, memory, and storage for administrative tasks. Recommended Specifications: Intel Core i5 processor or above, 8GB RAM or above, SSD storage, or workstations with similar or greater specifications.

Mobile Devices. Smartphones or tablets enable students and employers to access the system and scan QR codes. Recommended Specifications: Large screen, fast processor, built-in security features, or equivalent mobile devices that provide similar functionality.

Security Hardware. Security hardware should provide advanced threat intelligence capabilities and adhere to industry standards. Recommended Specifications: Palo Alto Networks PA-220 Next-Generation Firewall; YubiKey 5 Series hardware security keys, or other security hardware with equivalent capabilities.

Software Requirement

This section will provide the list of operations and activities that the system should perform and serve as an input to the design specification. This section will also serve as the guide for testing and validating the system.

Functional Requirements

The system will be designed to fulfill the following major functions:

1. Administering the system

2. Authorizing user’s access

3. Issuing of credentials / certificates

4. Verification of credentials / certificates

Table 6. *Functional Requirements*

ID	User	Requirement
FR001	Graduates	The system must have an interface for the graduates to register and receive their certificates.
FR002	Registrar	The system must have an interface for the school administrators.
FR003	Registrar	Through the administrator panel, the administrators can upload the digital certificates and other credentials of the graduates.
FR004	Employers	The system must have an interface for the employers to send inquiries for the verification of certificates.
FR005	Registrar	The system must have records and backups of all the transactions conducted.
FR006	Administrator	Manage system configurations, settings, and user roles.
FR007	Administrator	Oversee system maintenance, updates, and troubleshooting.
FR008	Registrar	Verify and authenticate users based on their graduate status.
FR009	Administrator	Manage user roles and permissions within the system.
FR010	Registrar	Enable the creation and issuance of digital credentials or certificates.
FR011	Registrar	Record and store issued credentials securely on the database.
FR012	Registrar	Provide a mechanism for third parties to verify the authenticity of credentials.
FR013	Administrator	Implement a secure process for revoking or invalidating credentials when necessary.
FR014	Graduates	Allow users to register profiles, providing necessary information for credential issuance.
FR015	Graduates	Enable users to manage and update their profile information.
FR016	Registrar	Ensure that all records are accurately and consistently stored.
FR017	Administrator	Allow administrators to enforce MFA for specific user roles or actions.
FR018	System Users (General)	The system must generate and deliver an OTP for MFA, verify it, and limit number of attempts.
FR019	Administrator	Create an audit trail to log and track system activities, changes, and user interactions.
FR020	Administrator	Enhance transparency and accountability through detailed system logs.
FR021	Administrator	Implement a notification system to keep users informed about credential issuance, changes, or system updates.
FR022	Administrator	Provide alerts for any actions requiring user attention.
FR023	Administrator	Generate reports and analytics to offer insights into system usage, verification trends, and other relevant metrics.
FR024	Administrator	Ensure interoperability and smooth information flow between systems.
FR025	Administrator	Establish a support system to assist users with inquiries, issues, or technical difficulties.
FR026	Administrator	Provide resources and assistance for users navigating the system.
FR027	Administrator	Implement advanced security features, such as two-factor authentication and encryption, to protect sensitive data.
FR028	Administrator	Develop a robust backup and recovery system to prevent data loss in case of unexpected events or system failures.
FR029	Administrator	Ensure data integrity and quick system restoration when needed.
FR030	Administrator	Ensure that the system complies with relevant data protection and privacy regulations.
FR031	Administrator	Stay updated on legal requirements and adapt the system accordingly.

Table 6 cont'd. *Functional Requirements*

FR032	Administrator	Incorporate a feedback system to gather user opinions and suggestions for continuous improvement.
FR033	System Users (General)	Use user feedback to enhance the overall user experience.
FR034	System Users (General)	Consider future growth and adaptability in system architecture.

Non-functional Requirements

1. Performance Requirement
    - The system should be able to handle a specified number of concurrent users without a significant decrease in performance. This requirement ensures that the system can efficiently manage multiple users simultaneously.
2. Scalability Requirement
    - The system architecture should be scalable to accommodate an increase in the number of users or data volume. This ensures that the system can grow to meet future demands without a major redesign.
3. Reliability and Fault Tolerance Requirement
    - The system should be designed to handle and recover gracefully from unexpected failures, minimizing downtime and maintaining continuous service availability.
4. Compliance Requirement

    - The system must adhere to relevant industry standards, legal regulations, and data protection laws. This ensures that the system operates within the boundaries of applicable rules and regulations.

5. Backup and Disaster Recovery Requirement
    - Regular backups of the system data should be performed, and a comprehensive disaster recovery plan should be in place to ensure data recovery and system restoration in the event of a catastrophic failure.
6. Monitoring and Reporting Requirement
    - Implement a robust system monitoring and reporting mechanism to provide administrators with real-time insights into the system's performance, usage patterns, and potential issues.
7. Compatibility Requirement
    - The system should be compatible with a variety of web browsers and devices to ensure a consistent user experience across different platforms.
8. Data Encryption Requirement
    - All sensitive data, including user credentials and transaction details, should be encrypted during transmission to ensure data security.
9. User Authentication Strength Requirement

   - Define the strength of user authentication mechanisms, such as password complexity requirements and multi-factor authentication, to enhance security.

10. System Logging Requirement
    - Extend the logging capabilities to capture detailed information about system events, errors, and warnings for diagnostic purposes.
11. Regulatory Compliance Requirement
    - Ensure that the system complies with industry-specific regulations and standards, particularly in the education sector.
12. User Training and Documentation Requirement
    - Develop comprehensive user documentation and provide training materials to facilitate user understanding and adoption of the system.
13. Mobile Responsiveness Requirement
    - Ensure that the system is responsive and provides an optimal user experience on mobile devices.
14. Feedback and Performance Improvement Requirement
    - Implement mechanisms to collect user feedback on system performance and features to inform continuous improvement efforts.
15. Cost Efficiency Requirement
    - Consider cost-effective solutions and optimizations to manage infrastructure and operational expenses.

Software Development Tools

This section discusses the essential tools used in developing the certificate authentication system.

Web Technologies for GUI

React.js: React.js is a JavaScript library for building dynamic and interactive user interfaces. It allows the development of reusable UI components, making web applications more efficient and maintainable. In this system, React.js will be used to create a responsive and interactive frontend, handling user interactions, rendering data dynamically, and improving overall user experience.

CSS: Cascading Style Sheets (CSS) control the presentation and layout of HTML elements. CSS will be employed to style the system, enhancing the visual appeal and supporting a user-friendly experience.

Bootstrap: Bootstrap is a front-end framework that includes pre-designed components and a responsive grid system. It simplifies the development of responsive and mobile-first web pages, making that the system is accessible and looks professional on various devices, including desktops, tablets, and smartphones.

Backend Programming Languages

Python: Python is chosen for backend development due to its simplicity, readability, and extensive library support. It handles core functionalities such as certificate generation, encryption, decryption, and user authentication. Python's rich ecosystem includes libraries for cryptography and web development, making it an ideal choice for developing the system’s backend.

Encryption Libraries

Python's Cryptography Library: This library provides robust and secure methods for cryptographic operations, including ECC and digital signatures. It will be used to encrypt certificate data and verify its authenticity, maintaining the security of the information.

QR Code Generation

qrcode Python Package: The qrcode library allows for the easy generation of QR codes. QR codes will be used to encode encrypted certificate data or URLs pointing to stored encrypted data, enabling quick and convenient verification by scanning the codes.

Development Environment

Visual Studio Code (VS Code): VS Code is a powerful code editor that supports a wide range of programming languages and frameworks. It provides features such as syntax highlighting, debugging, and version control integration. VS Code will be used as the primary development environment for writing and managing the system’s codebase.

Database

MySQL: MySQL is a reliable and open-source relational database management system. It will be used to securely store encrypted certificates, user data, and other related information. MySQL's performance and ease of use make it suitable for handling the data requirements of the system.

Web Framework

Django: Django is a high-level Python web framework that encourages rapid development and clean design. It provides tools for handling user authentication, URL routing, and database interactions. In the system development, Django will be used to develop the web application, allowing both frontend and backend functionalities are managed effectively. Django powers the core application logic, including:

- User authentication – Grants access to certificate data exclusively to authorized users.
- Secure data handling – Managing encrypted certificates and verification processes.
- Dynamic page rendering – Allowing users to apply for, issue, and verify certificates in real-time.

Multi-Factor Authentication (MFA) with TOTP

PyOTP: PyOTP is a Python library for generating and verifying Time-based One-Time Passwords (TOTP). It will be used to implement TOTP for Multi-Factor Authentication (MFA) via email. PyOTP will help ensure that users provide a time-sensitive code in addition to their regular credentials, enhancing the security of the system.

Email Libraries (e.g., smtplib): Python's smtplib or other email libraries will be used to send TOTP codes to users via email. This will enable the MFA process, where users receive a one-time password to their email address to verify their identity during login or other critical operations.

System Architecture

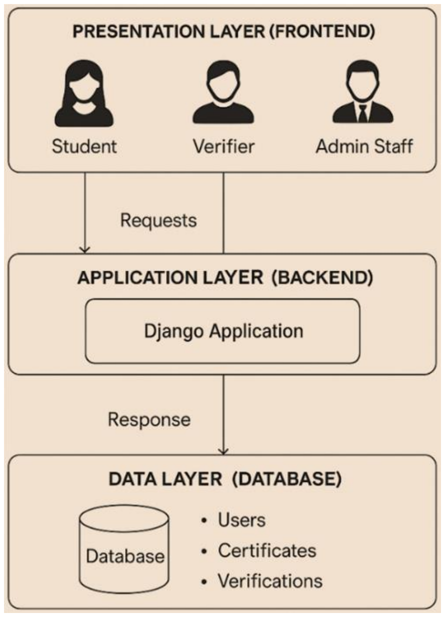


Figure 4. *System Architecture*

The system architecture for the certificate verification platform is structured into three distinct layers: the Presentation Layer, the Application Layer, and the Data Layer. Each layer plays an important role in supporting the system's functionality, security, and scalability.

The Presentation Layer serves as the user interface, allowing different types of users such as students, verifiers, and administrative staff to interact with the system. Students can upload and view certificates, verifiers can scan QR codes to validate certificates, and administrators can manage certificate issuance and user records. This layer is responsible for sending requests to the backend and displaying responses, providing a smooth and user-friendly experience.

The Application Layer, powered by a Django backend, handles the core logic of the system. It processes requests from the frontend and performs essential operations such as hashing certificate data using SHA-256, generating ECC key pairs, creating digital signatures, and producing Zero-Knowledge Proofs (ZKP). It also manages the verification process by recomputing ZKP proofs, comparing them with those extracted from QR codes, and validating digital signatures using public keys. This layer ensures that all cryptographic and business logic is securely and efficiently executed.

The Data Layer consists of the database that stores all persistent information. This includes user profiles, certificate records, ZKP commitments, digital signatures, public keys, and verification logs. The backend interacts with this layer to retrieve and update data as needed during issuance and verification processes. By maintaining a well-structured and secure data repository, this layer supports the integrity and reliability of the entire system. Together, these three layers form a modular and scalable architecture that supports secure certificate issuance and verification using advanced cryptographic techniques.

System Workflow Diagram

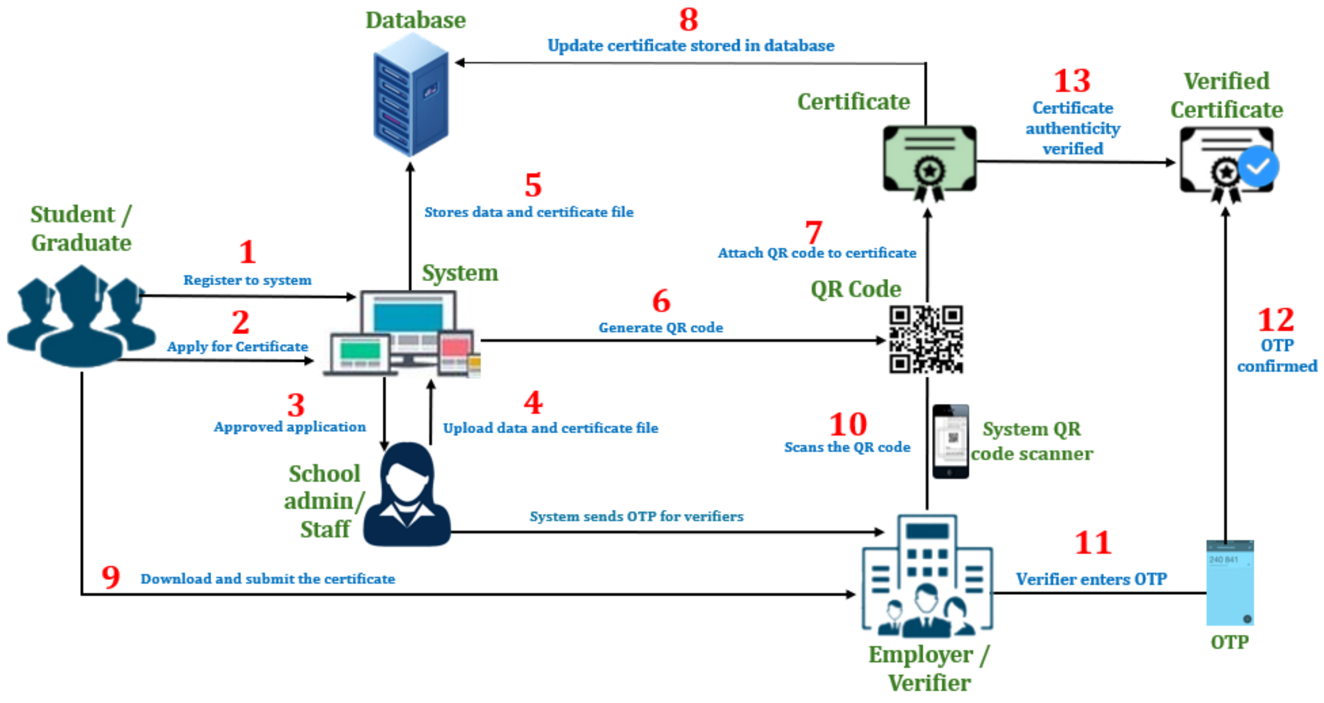


Figure 5. *System Workflow Diagram*

The certificate authentication process begins with students or graduates registering in the system. Once registered, they apply for a certificate through the system’s interface. The school admin or staff reviews the application, and upon approval, they upload the certificate data and relevant files into the system. The system then stores this data securely in the database.

Next, the system generates a QR code, which is attached to the certificate before being stored in the database as an updated certificate record. Graduates can then download their certificates, which contain the embedded QR code. Employers or verifiers who receive the certificate can scan the QR code using a system QR code scanner. Once scanned, the system sends a one-time password (OTP) to the verifier. The verifier must enter the OTP into the system to confirm their identity.

Upon successful OTP confirmation, the system verifies the authenticity of the certificate. If the verification process is successful, the system confirms that the certificate is valid and has not been tampered with. This structured process ensures that certificates are securely issued, stored, and verified, preventing fraudulent alterations and unauthorized access.

Use Case

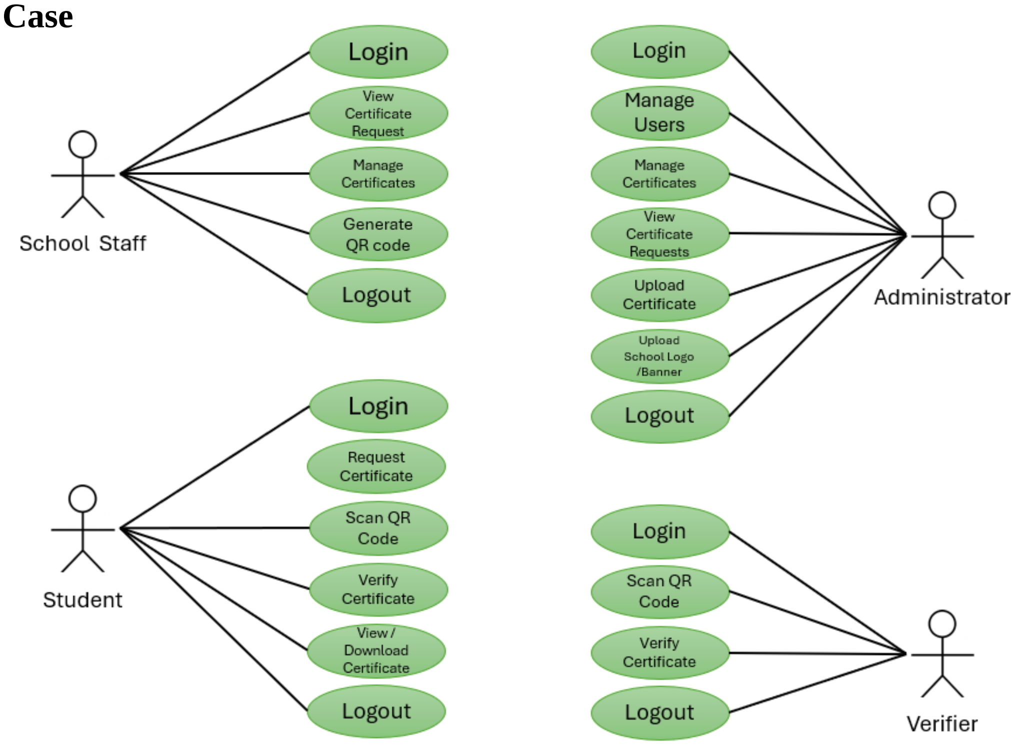


Figure 6. *Use Case*

The system consists of four main user roles: Administrator, School Staff, Student, and Verifier. Each role has specific functions within the certificate authentication system.

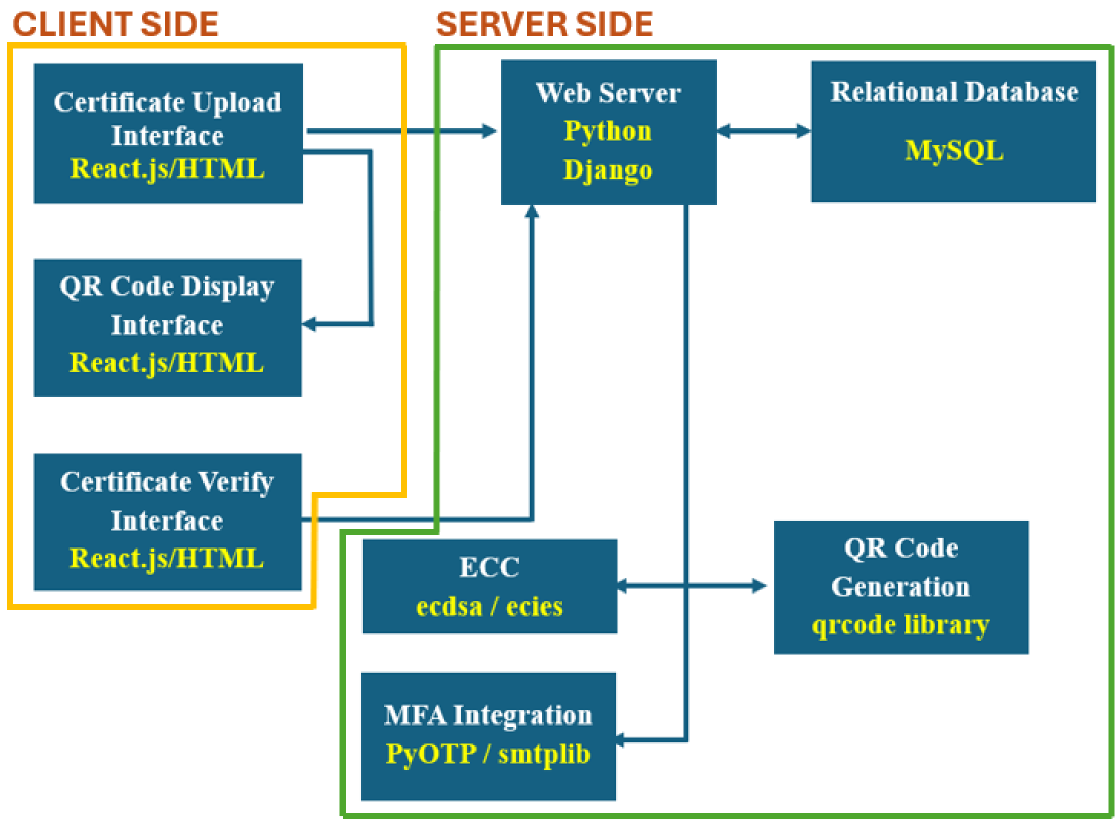
The Administrator has the highest level of control, managing user accounts and overseeing certificate-related tasks. They can manage users, approve or reject certificate requests, upload certificates, and modify the school logo and banner. The administrator ensures that the system operates smoothly and that only authorized users can access key functionalities.

The School Staff process certificate requests. They can view certificate requests submitted by students, manage certificates, and generate QR codes which are embedded in the certificates before being issued. Once the certificates are prepared and verified, they are uploaded to the system for students to download.

The students can log in to request a certificate, scan QR codes, verify certificates, and download issued certificates. Once their request is processed and approved, they can access their digital certificate, giving them a secure and verifiable copy for academic or employment purposes.

The Verifier, typically an employer, can scan QR codes and verify certificates through the system. When a QR code is scanned, the system automatically checks its authenticity and confirms whether the certificate is valid and issued by the institution.

Client – Server Model



CLIENT SIDE
SERVER SIDE
Certificate Upload Interface React.js/HTML
QR Code Display Interface React.js/HTML
Certificate Verify Interface React.js/HTML
Web Server Python Django
Relational Database MySQL
ECC ecdsa / ecies
QR Code Generation qrcode library
MFA Integration PyOTP / smtplib

Figure 7. *Client – Server Model*

The integration of various client-side and server-side components is essential for delivering a seamless and secure process. On the client side, the system begins with the Certificate Upload Interface developed using React.js and HTML. This interface provides users with a web-based form to upload their certificates. Once a certificate is uploaded, the interface sends the data to the server. The user experience is straightforward, enabling easy submission of certificate files.

Following the upload, the server-side components take over. The web server, built with Python Django, receives the uploaded certificate data. The server processes the data, utilizing ECC encryption via the *ecdsa* library to secure the certificate information. The encrypted certificate data is then stored in a MySQL relational database.

After encryption, the server generates a QR code that encapsulates the encrypted certificate data. This QR code generation is facilitated by integrating services using email. The generated QR code is then sent back to the client-side interface, where it is displayed in the QR Code Display Interface. Users can download or print this QR code for future use.

For verification purposes, the system includes a Certificate Verify Interface, also developed using React.js and HTML. Users can scan the QR code through this interface, which then sends the scanned data back to the server. The server decrypts the QR code data using ECC decryption, retrieves the original certificate from the database, and verifies its authenticity. The verification results are then sent back to the client-side interface and displayed to the user, completing the verification process.

QR Code Structure and Generation Process

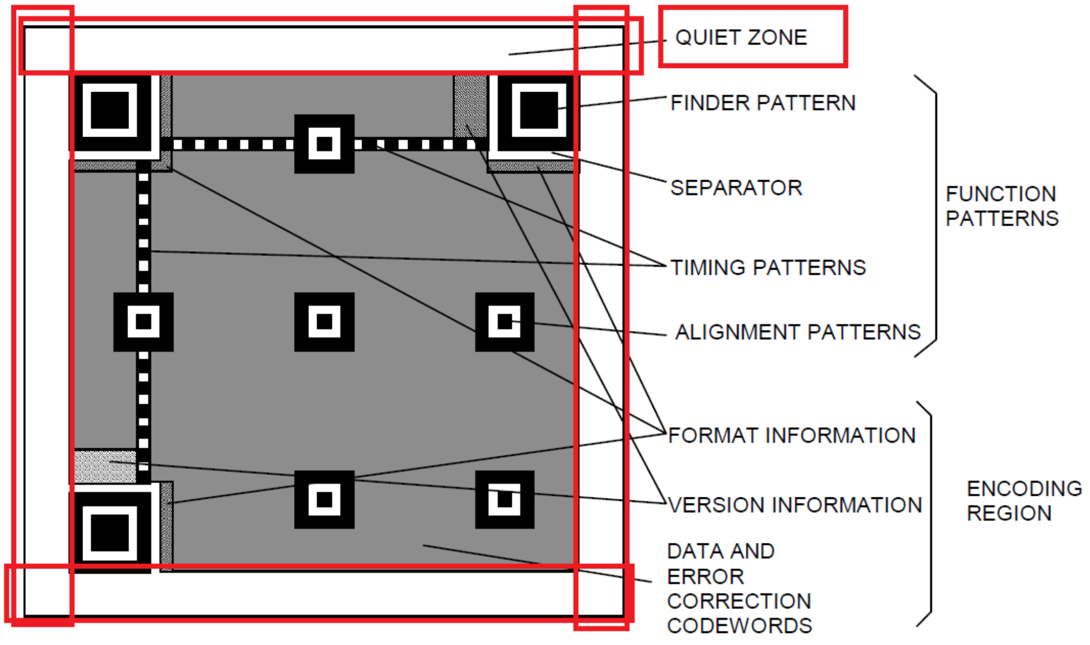


Figure 8. *QR Code Structure*

The QR code structure in the system includes several key components that are essential for the functionality and readability of the QR code. These components will form the structured matrix that represents the visual pattern of the QR code, enabling it to be scanned and the encrypted data to be decoded and accessed securely. These components are:

1. Version Information: This specifies the size and data capacity of the QR code. QR codes have different versions, which correspond to different sizes and data capacities.
2. Format Information: This indicates the error correction level and data mask pattern used in the QR code. The error correction level determines how much

data can be restored if the QR code is damaged, while the data mask pattern is used to enhance the readability of the QR code.

3. Data and Error Correction Areas: These areas contain the encrypted data and error correction data. The error correction data is generated using the Reed-Solomon algorithm and allows the QR code to be read even if it is partially damaged.
4. Position Detection Patterns: These are special patterns located at the corners of the QR code that help the scanner identify the position and orientation of the QR code. This allows the QR code to be read from any direction.
5. Alignment Patterns: These are additional patterns found in larger QR codes that assist in navigating and reading the code accurately.
6. Timing Patterns: These patterns are used to determine the size of the modules within the QR code, which enables accurate interpretation of the code's structure by the scanner.
7. Quiet Zone: This is a margin around the QR code that does not contain any data modules. The quiet zone is necessary to ensure that the scanner can distinguish the boundaries of the QR code and read it properly.

The system is designed to generate QR codes that securely store and share confidential information. The process begins with data encryption, where certificate details are encrypted using the institution’s ECC public key, so that only authorized verifiers with the corresponding ECC private key can decrypt the information. Once encrypted, the data is formatted into a byte string containing QR header information, error correction bytes, and masking elements which supports proper structuring for QR code generation. This byte string is then transformed into a 2D matrix, which visually represents the QR code and includes essential components such as version and format information, data and error correction areas, position detection patterns, alignment patterns, timing patterns, and a quiet zone to maintain layout integrity.

To enhance reliability, error correction is applied, allowing the QR code to be reconstructed even if parts of it are damaged or missing. The system further ensures authenticity by generating a SHA-256 hash of the certificate data, which is digitally signed using the institution's ECC private key (ECDSA), preventing forgery and unauthorized modifications. Finally, the matrix is synthesized into an image format, embedding the ECC-encrypted data, hash value, and digital signature into the QR code, making it ready for printing or digital sharing.

The system is implemented with a frontend built using React.js, HTML, CSS, and JavaScript for mobile responsiveness, while the backend is developed with Python Flask/Django**,** ensuring secure processing of encryption, QR code generation, and verification. A MySQL database is utilized for storing certificate data, ECC keys, and verification logs. Security measures such as ECC encryption, digital signatures, and Multi-Factor Authentication (MFA) during verification provide an added layer of protection. This system offers several advantages, including strong security through ECC encryption, compact QR code generation, robust error correction, and flexible deployment, allowing QR codes to be printed or shared digitally for seamless verification.

ECC Public Key and Private Key

This system employs the Elliptic Curve Digital Signature Algorithm (ECDSA). The private key, securely held by the issuing authority, is used to digitally sign certificates, preserving the integrity of the certificates after issuance. Since the private key is confidential, only the authorized entity can generate valid certificates, significantly reducing the risk of forgery.

During the verification process, the system securely retrieves and utilizes the necessary cryptographic information to validate the certificate without exposing sensitive keys. If the certificate has been modified, the verification will fail, signaling potential tampering.

By implementing ECDSA in a secure and controlled manner, the system ensures that only certificates issued by a trusted authority are considered valid. This approach strengthens certificate security, prevents unauthorized modifications, and streamlines the verification process through QR code scanning. The use of digital signatures significantly reduces the risks of certificate forgery and unauthorized alterations, reinforcing trust in the certificate verification process.

Function	**Public Key**	**Private Key**
Digital Signature	Verify digital signature (created with the private key using ECDSA).	Sign data or certificates (create digital signature using ECDSA).

Table 7. *Roles of ECC Public Key and Private Key*

Process Flowchart

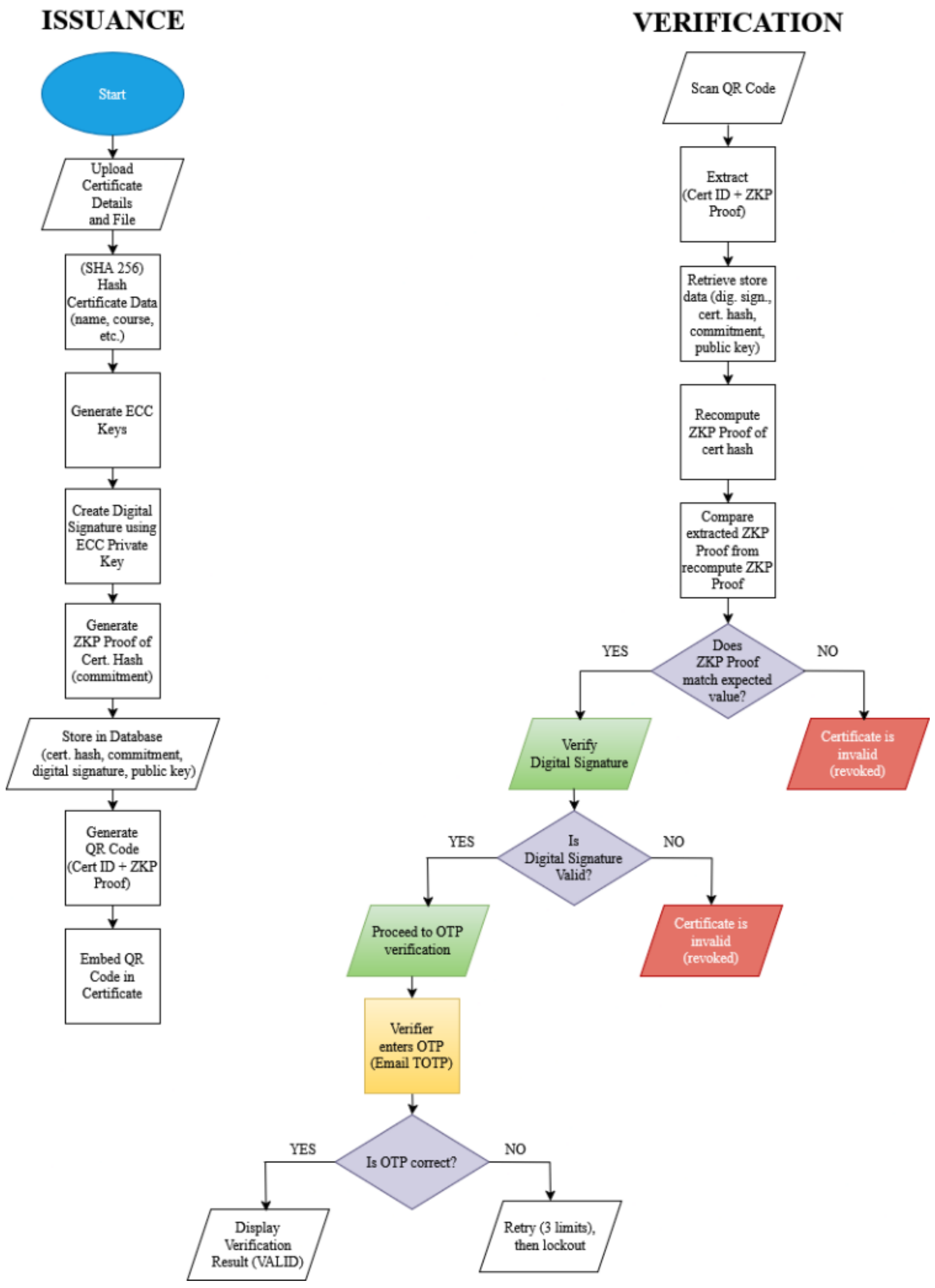

ISSUANCE
Start
Upload Certificate Details and File
(SHA 256) Hash Certificate Data (name, course, etc.)
Generate ECC Keys
Create Digital Signature using ECC Private Key
Generate ZKP Proof of Cert. Hash (commitment)
Store in Database (cert. hash, commitment, digital signature, public key)
Generate QR Code (Cert ID + ZKP Proof)
Embed QR Code in Certificate
VERIFICATION
Scan QR Code
Extract (Cert ID + ZKP Proof)
Retrieve store data (dig. sign., cert. hash, commitment, public key)
Recompute ZKP Proof of cert hash
Compare extracted ZKP Proof from recompute ZKP Proof
Does ZKP Proof match expected value?
YES
NO
Verify Digital Signature
Certificate is invalid (revoked)
Is Digital Signature Valid?
YES
NO
Proceed to OTP verification
Certificate is invalid (revoked)
Verifier enters OTP (Email TOTP)
Is OTP correct?
YES
NO
Display Verification Result (VALID)
Retry (3 limits), then lockout

Figure 9. *System Flowchart*

The system begins with the certificate issuer uploading the certificate details and file. These details such as the recipient's name, course title, and completion date are hashed using the SHA-256 algorithm to produce a unique digital fingerprint of the certificate data. Next, the system generates a pair of ECC (Elliptic Curve Cryptography) keys: a private key and a public key. The private key is used to create a digital signature from the hashed certificate data to ensure that the certificate can later be verified as coming from a trusted source.

Zero-Knowledge Proof (ZKP) is generated from the certificate hash for data integrity and privacy preservation. This ZKP proof allows verifiers to confirm that the certificate data is valid without revealing the actual contents. The certificate hash, ZKP commitment, digital signature, and public key are stored in a secured database. A QR code is generated containing the certificate ID and the ZKP proof, which is embedded into the certificate itself.

During verification, the QR code is scanned to extract the certificate ID and the original ZKP proof. The system retrieves the stored certificate hash, digital signature, and public key from the database. ZKP proof will be recomputed from the stored hash and compares it with the ZKP proof extracted from the QR code. If the proofs match, it confirms that the certificate data has not been tampered with. The system uses the retrieved public key to verify the digital signature, confirming that the certificate was indeed issued by the legitimate authority. If both the ZKP and digital signature are valid, the system may proceed with OTP (One-Time Password) verification step before finally displaying the result as a valid certificate. Table 8 presents an example of process in structured, side-by-side format for both issuance and verification.

Table 8. *Example Scenario of Certificate Issuance and Verification*

Issuance Process	**Verification Process**
The system hashes the certificate data: "Ana Cruz | Blockchain Fundamentals | August 15, 2025" → Produces a unique SHA-256 hash.	The verifier scans the QR code and extracts: - Certificate ID - ZKP proof
An ECC key pair is generated: - Private Key: Used to sign the hash. - Public Key: Used later for verification.	The system retrieves from the database: Certificate hash, Digital signature, and public key
A digital signature is created using the private key and the hash.	The ZKP proof is recomputed from the retrieved hash.
A Zero-Knowledge Proof (ZKP) is generated from the hash to allow verification without exposing the actual data.	The recomputed ZKP proof is compared with the one from the QR code: If they match → Data is intact. If they don't match → Certificate is invalid or tampered.
The following are stored in the database: - Certificate hash - ZKP proof (commitment) - Digital signature - Public key	The system uses the public key to verify the digital signature: If valid → Confirms authenticity of the issuer. If invalid → Certificate is rejected.
A QR code is generated containing: Certificate ID and ZKP proof	Optional: OTP verification may be required for added security.
The QR code is embedded into Ana's certificate.	Result: Valid Certificate if all checks passed.

Table 9 presents the process in actual usage, showing both user experience and the back-end logic

Table 9. *Real world scenario and back -end operations*

Real-World Process	**Backend Operations**
Ana completes a course and requests a certificate.	System receives and stores certificate details.
Admin inputs Ana's data into the system.	System hashes the data using SHA-256.
Certificate is prepared and sent.	ECC keys are generated; hash is signed with private key.
QR code is added to the certificate.	ZKP proof is generated and stored with other data.
Employer scans the QR code.	System extracts certificate ID and ZKP proof.
Verification is requested.	System retrieves hash, signature, and public key.
System checks certificate validity.	ZKP is recomputed and compared with QR code proof.
Certificate is confirmed authentic.	Signature is verified using the public key.

Database Schema Design

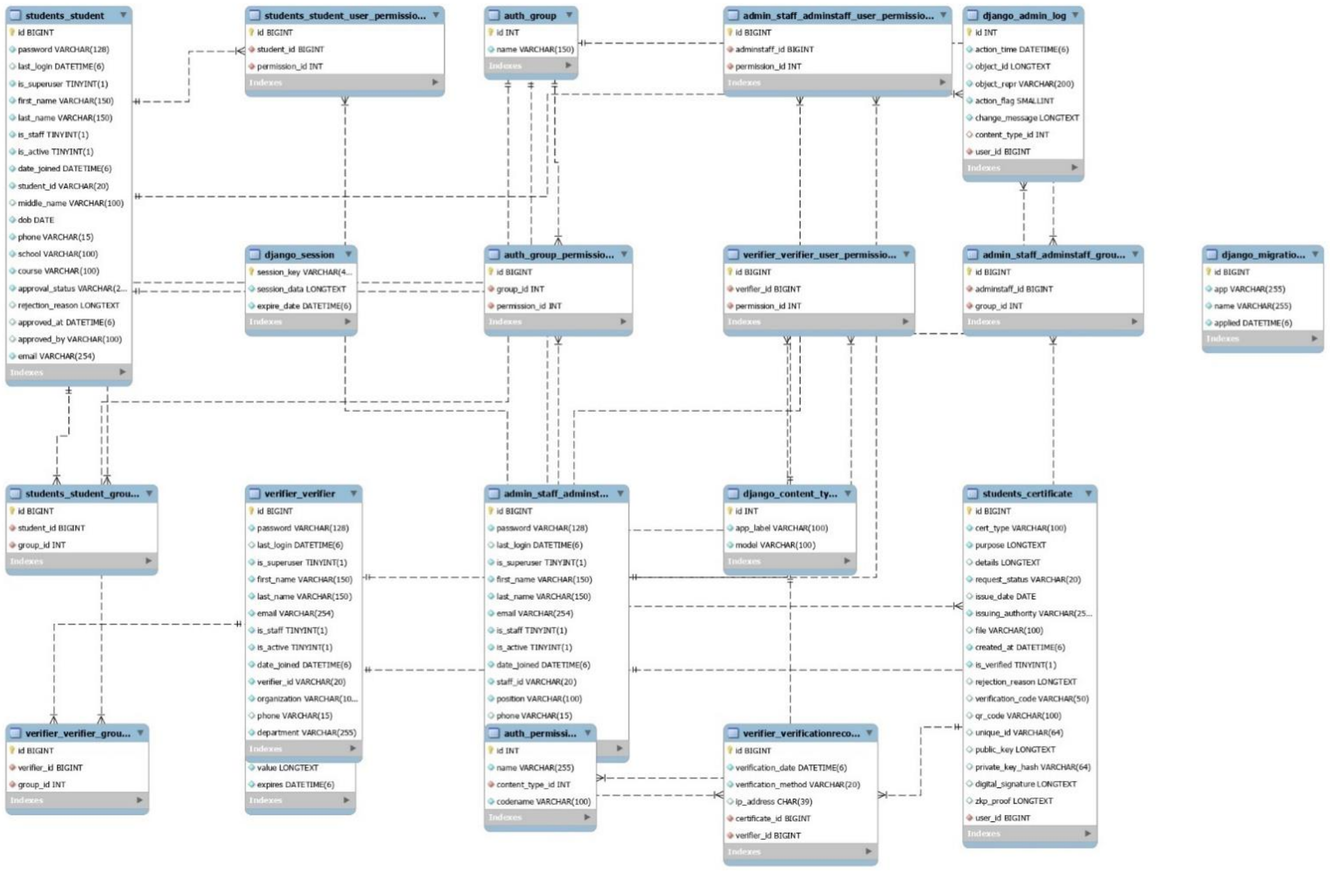


Figure 10. *Database Schema*

The database schema illustrates a role-based system for managing and verifying certificates. It is structured around three main user roles: students, verifiers, and admin staff, each represented by its own table with associated permissions and group affiliations. The design appears to follow Django's built-in user and permission management system, allowing for scalable and secure role-based access control.

At the center of the system is the *students_certificate* table, which holds detailed information about each certificate issued. This includes fields for certificate name, purpose, issue date, issuing authority, verification code, a unique identifier, and supporting data such as a QR code, digital signature, and status fields to track whether a certificate has been verified or rejected. Each certificate is linked to a student and optionally approved by a user, allowing for accountability in the certificate issuance process.

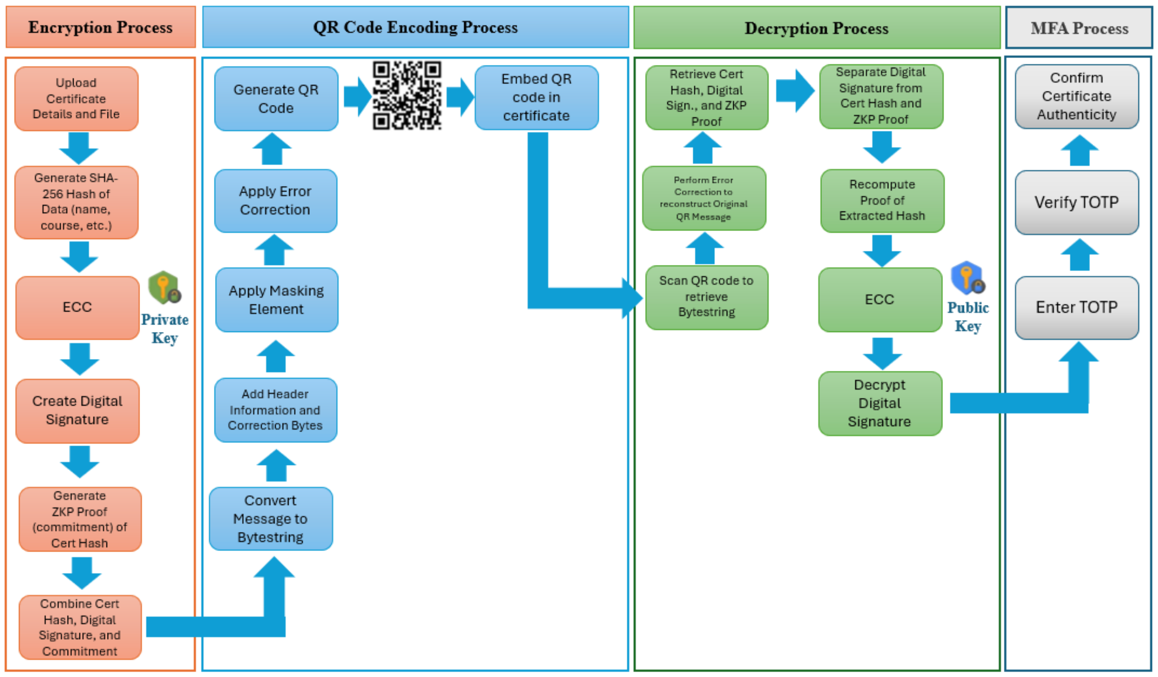
The *students_student* table manages student profiles, storing personal data such as names, date of birth, contact information, and their current course and school. Status fields like *approval_status* and *approved_at* indicate the state of each student's certificate request or enrollment status. Verifiers are managed through the *verifier_verifier* table, which includes professional details such as organization, department, and email, as well as account-related data like login timestamps and status.

Administrative users are managed in the *admin_staff_adminstaff* table. This table mirrors the structure of the other user roles but includes additional fields like position and a flag indicating staff status. These admin users also have dedicated group and permission tables, helping to enforce access restrictions based on administrative roles.

Verification activities are logged in the *verifier_verificationrecord* table, which stores when a certificate was verified, by whom, and includes metadata like the IP address used. This provides a clear audit trail for verification events.

Supporting tables such as *auth_group*, *auth_permission*, and various *user_permissions* and *group* tables allow for flexible role and permission management. Other Django system tables like *django_session*, *django_admin_log*, *django_migrations*, and django_content_type are also present to support session handling, logging, migrations, and content type referencing.

Security Model and Encryption Workflow



Encryption Process
QR Code Encoding Process
Decryption Process
MFA Process
Upload Certificate Details and File
Generate SHA-256 Hash of Data (name, course, etc.)
ECC
Private Key
Create Digital Signature
Generate ZKP Proof (commitment) of Cert Hash
Combine Cert Hash, Digital Signature, and Commitment
Convert Message to Bytestring
Add Header Information and Correction Bytes
Apply Masking Element
Apply Error Correction
Generate QR Code
Embed QR code in certificate
Scan QR code to retrieve Bytestring
Perform Error Correction to reconstruct Original QR Message
Retrieve Cert Hash, Digital Sign., and ZKP Proof
Separate Digital Signature from Cert Hash and ZKP Proof
Recompute Proof of Extracted Hash
ECC
Public Key
Decrypt Digital Signature
Enter TOTP
Verify TOTP
Confirm Certificate Authenticity

Figure 11. *Security Model and Encryption Workflow*

The certificate authentication and verification process begin when the institution manually prepares and processes the student's certificate, which includes essential information such as the student's name, course or degree, and date of graduation. Once finalized, the certificate file is uploaded through the system's GUI. During the encryption stage, the system generates a SHA-256 hash of the certificate's key information, which serves as a digital fingerprint of the document. This hash is then signed using the Elliptic Curve Digital Signature Algorithm

(ECDSA) with the institution’s ECC private key to ensure the document’s integrity and authenticity. To add a privacy-preserving layer of security, the system also generates a Zero-Knowledge Proof (ZKP) commitment that allows the verifier to confirm the validity of the certificate without exposing the actual hash or private key. The certificate hash, digital signature, and ZKP commitment are then combined into a structured message and converted into a byte sequence. Before encoding, header information and error correction bytes are added to support data recovery, and a masking pattern is applied to enhance QR code readability and robustness. The final QR code is generated from the encoded message and embedded into the certificate. When verification is initiated, a verifier scans the QR code using the system’s tool, which decodes the bytes and applies error correction if needed. The system retrieves and separates the digital signature, certificate hash, and ZKP proof. It then recomputes the proof of the extracted hash and verifies the digital signature using the ECC public key of the issuing institution. A comparison is made between the recomputed and extracted hashes to detect any tampering. For added security, Multi-Factor Authentication (MFA) is employed—requiring the verifier to enter a Time-Based One-Time Password (TOTP) sent to their registered email. The system verifies the TOTP, and upon successful validation, it confirms the authenticity of the certificate. This multi-layered approach ensures secure, verifiable, and tamper-resistant certificate authentication using QR codes, RSA/ECC, ZKP, and MFA.

Secured Transmission Using TLS

To ensure the confidentiality and integrity of data transmitted between users and the server, the system employs Transport Layer Security (TLS) as the underlying secure communication protocol. TLS provides end-to-end encryption during data exchanges, preventing eavesdropping and tampering by third parties. All traffic occurs over HTTPS, which layers TLS on top of HTTP to establish a secure session.

The comparison between unencrypted (HTTP) and encrypted (HTTPS) communication is illustrated in the figure below. In a plaintext connection (HTTP), sensitive user data such as usernames and passwords are transmitted in readable format, posing a major security risk. Conversely, HTTPS encrypts these details, rendering intercepted data unreadable.

In the context of this system, TLS secures various key interactions:

- **Certificate File Uploads**

  When administrative staff upload certificate files through the web interface, the data is encrypted in transit, preventing unauthorized parties from intercepting or altering the document before it reaches the server.

- **User Login Sessions**

  TLS encrypts login credentials (usernames and passwords) for students, staff, and verifiers. This prevents credential theft via packet sniffing or man-in-the-middle attacks.

- **TOTP Verification Exchanges via Email**

Although OTP codes are sent via SMTP (email), TLS secures the web-based verification process when a user submits the TOTP through the system’s form. This protects against interception during submission.

- **Verifier Requests (QR Scan & Record Fetch)**

  When employers or authorized verifiers scan a QR code and initiate a verification request, the communication between their browser and the server where certificate details and digital signatures are validated—is protected through TLS. This ensures the certificate data and verification results are not visible to intermediaries.

By enforcing TLS across all sensitive communication endpoints, the system upholds privacy, resists interception, and maintains a secure environment for handling credentials and verification routines.

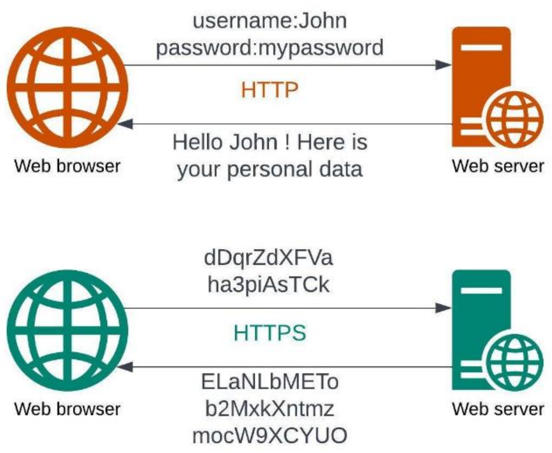


Figure 12. *TLS-Secured Communication of the System*

Module Implementation (Technology-based modules)

This study introduces an enhanced authentication model, which is the core contribution of the system. The model integrates QR code encryption, ECC-based signing, Zero-Knowledge Proofs (ZKP), and Multi-Factor Authentication (MFA) into a unified framework that strengthens certificate authentication. Unlike traditional systems that focus only on QR code storage or ECC signing, this model provides multi-layered protection and privacy-preserving verification. In this study, academic credentials serve only as the test dataset to demonstrate the model’s effectiveness. The focus is not merely on certificate management but on the design of a scalable authentication framework applicable to other domains requiring secure document and identity verification. The model is structured into four integrated layers:

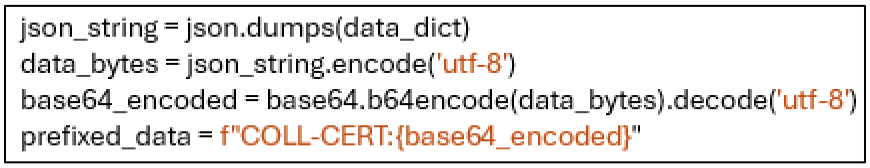
1. Data Encoding Layer – where certificate details are hashed and signed using ECC.
2. QR Code Security Layer – where encrypted and signed data is embedded in tamper-resistant QR codes.
3. Cryptographic Proof Layer – where ZKP ensures authenticity without disclosing sensitive values.
4. User Authentication Layer – where MFA via email-based OTP prevents unauthorized access.

This section details the core modules developed in the system and explains how each was implemented, including encryption, signature generation, QR encoding, and cryptographic proof generation. All modules are implemented using Python and cryptographic libraries and work together to provide a secure certificate authentication process.

QR Code Generation Module

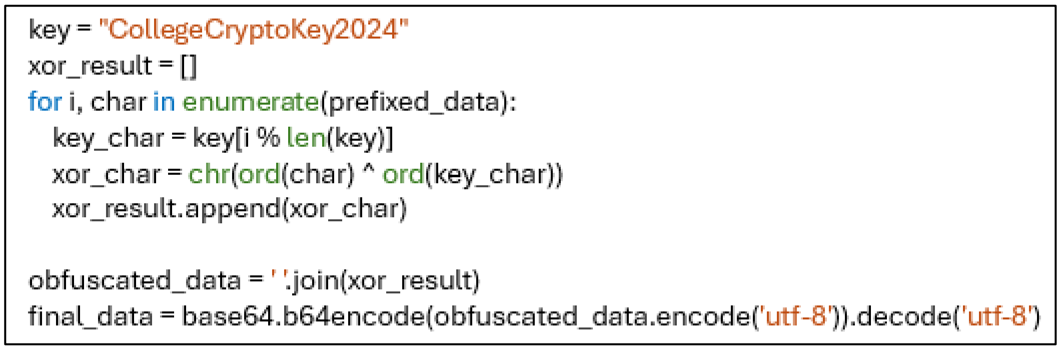
The system uses the qrcode Python library to generate QR codes that carry encrypted certificate data. The QR codes are not readable by standard scanners because the data is first serialized into JSON format and then encrypted. Encryption is performed by first converting the data into a Base64 string and prefixing it with COLL-CERT:. A static XOR key (CollegeCryptoKey2024) is then used to obfuscate the string before it is re-encoded in Base64 and passed into the QR generator.

Example:



```
json_string = json.dumps(data_dict)
data_bytes = json_string.encode('utf-8')
base64_encoded = base64.b64encode(data_bytes).decode('utf-8')
prefixed_data = f"COLL-CERT:{base64_encoded}"
```

Then, XOR encryption is applied, and the encrypted string is encoded again into Base64:



```
key = "CollegeCryptoKey2024"
xor_result = []
for i, char in enumerate(prefixed_data):
  key_char = key[i % len(key)]
  xor_char = chr(ord(char) ^ ord(key_char))
  xor_result.append(xor_char)

obfuscated_data = ' '.join(xor_result)
final_data = base64.b64encode(obfuscated_data.encode('utf-8')).decode('utf-8')
```

This final_data is embedded in the QR code using the qrcode library and then linked to the certificate for download.

ECC-Based Encryption Module

Elliptic Curve Cryptography (ECC) is used to generate the digital signature that ensures data authenticity. The system signs a SHA-256 hash of the certificate content using ECDSA with the institution’s private key.

Implementation Logic:

- The data is first serialized in sorted JSON format.
- The ECDSA key is loaded using the ecdsa library.
- The data is signed and encoded using DER format.

Example:

```
sk = SigningKey.from_pem(private_key_pem)

signature = sk.sign(certificate_data.encode(), sigencode=sigencode_der)
```

The signed data includes fields such as cert_id, student_id, student_name, cert_type, and a timestamp. The public key is stored in PEM format and used later for verification.

Multi-Factor Authentication Module

This module provides additional verification using a Time-based One-Time Password (TOTP) sent via email. When a verifier attempts to check a certificate, the system sends an OTP to their registered email using Django’s built-in email system. The OTP is cached and is valid for 5 minutes.

OTP Generation:

```
def generate_otp(length=6):

  return ''.join(random.choice(string.digits) for _ in range(length))
```

Email Sending:

```
send_mail(
  subject,
  message,
  settings.DEFAULT_FROM_EMAIL,
  [email],
  fail_silently=False,
)
```

The OTP must match the cached value for the verification to complete. If the OTP is incorrect or expired, access is denied.

Zero-Knowledge Proof Module

The ZKP module implements a Schnorr protocol using ECC. This allows the system to prove that the holder possesses the private key used for signing, without revealing the key itself. ZKPs are embedded directly in the QR code and are not stored in the database.

ZKP Components Stored in the QR Code:

```
"zkp": {
  "R": "Base64 commitment point",
  "e": "Hex challenge",
  "s": "Hex response"
}
```

Verification Process:

- Decode R, e, and s.
- Recompute $R' = g^s + Y^e$, where ***Y*** is the public key.
- Compare the recomputed ***R'*** with the original commitment.

Example:

```
gs_point = g * s
ye_point = pubkey_point * e
computed_R = gs_point + ye_point
```

If the recomputed and original points match, ZKP verification passes.

Testing and Evaluation

The following tests will be conducted to verify and validate if the system satisfies the objectives of the study:

Unit Integration Testing

This section outlines unit-level and integration-level testing approaches. Unit testing focuses on verifying the correctness of individual components such as QR code generation, ECC key creation, MFA handling, and ZKP logic. Integration testing, on the other hand, ensures that modules interact correctly, particularly between the frontend, backend logic, cryptographic services, and database operations. Testing will be conducted in a controlled development environment prior to system deployment. To ensure real-world validation, actual testing will be performed by designated individuals, including:

- System developers, to verify component logic and module integration.
- IT staff, to simulate realistic usage scenarios and uncover interaction-level issues.
- End users (such as school registrars, verifiers, and graduating students), to evaluate usability and consistency of outputs.

Each test case will be documented, and the results will inform final refinements before release. The results column will be populated during the actual testing phase in Chapter IV.

Table 10. *Unit and Integration Testing*

Test Case	**Test Level**	**Target Module**	**Test Description**	**Expected Outcome**	**Results**
QR Generation with Valid Certificate	Unit Test	QR Code Module	Generate a QR code for a valid certificate	QR is successfully generated with correct encrypted data	
QR Parsing and Payload Extraction	Unit Test	QR Code Module	Scan QR and parse the payload	Parsed data matches the original input	
ECC Key Pair Generation	Unit Test	ECC Module	Generate ECC public-private key pair	Keys are created correctly and stored securely	
ECC Encryption of Certificate	Unit Test	ECC Module	Encrypt sample certificate data	Ciphertext is not readable and decrypts correctly	
ECC Decryption of Encrypted Payload	Unit Test	ECC Module	Decrypt QR payload using ECC	Output matches original certificate	
MFA Code Generation	Unit Test	MFA Module	Generate OTP code for an email-authenticated user	Code is generated and valid for time window	
MFA Code Verification	Unit Test	MFA Module	Verify entered OTP against system-generated one	Verification succeeds or fails appropriately	
ZKP Proof Creation	Unit Test	ZKP Module	Generate proof of certificate ownership without revealing full data	Proof values generated successfully	
ZKP Verification	Unit Test	ZKP Module	Submit proof to verifier	Verification passes for correct proof, fails otherwise	
Certificate Request Processing	Unit Test	Certificate Module	Submit new certificate request	System stores request and triggers workflow	
Certificate Issuance	Unit Test	Admin/Registrar Module	Approve and issue certificate	Certificate generated and QR is attached	
Verify Certificate Using QR	Integration Test	QR + ECC + ZKP Modules	Scan QR, decrypt payload, validate hash, and verify ZKP	System verifies certificate without exposing full details	
User Login with MFA	Integration Test	Login + MFA Module	Login using username/password + OTP	Access granted after both credentials and OTP are correct	

Table 10 cont'd. *Unit and Integration Testing*

Admin/Staff User Role Functionality	Integration Test	Dashboard + DB Modules	Access management dashboard and view issued certs	Roles access only allowed functions	
Certificate Database Read/Write	Integration Test	Backend + Database	Insert and retrieve certificate records		
Verifier Certificate Lookup	Integration Test	Verifier + Backend	External verifier scans QR, system retrieves and verifies info		

Security Testing

This section outlines the planned security testing procedures for the major modules of the system, particularly those involving cryptographic operations and authentication mechanisms. The goal is to validate the integrity, resilience, and reliability of the system under simulated attack scenarios and failure conditions. Testing will include hash mismatch simulations, invalid QR payload processing, and multi-factor authentication (MFA) failure tests to ensure the system can securely handle expected and unexpected inputs. The results column will be populated during the actual testing phase in Chapter IV.

Table 11. *Security Testing*

Test Case	**Target Module**	**Test Description**	**Expected Outcome**	**Results**
Invalid QR Payload Format	QR Code Module	Input a tampered or malformed QR code	System should reject the code and display an error message	
Expired QR Code	QR Code Module	Use a QR code beyond its validity period	System should deny verification and inform the user	
Mismatched ECC Public Key	ECC Module	Decrypt with incorrect public key	Decryption should fail or produce invalid content	

Table 11 cont'd. *Security Testing*

Simulated Hash Tampering	ECC/QR Module	Modify the hash value inside the QR payload	System should detect tampering and halt the process	
MFA OTP Input Failure	MFA Module	Input wrong OTP multiple times	Account should be temporarily locked or flagged	
OTP Replay Attack Simulation	MFA Module	Reuse a previously valid OTP	System should detect and deny the attempt	
Incomplete ZKP Proof Submission	ZKP Module	Submit partial proof values	System should reject the incomplete proof	
Invalid ZKP Proof Simulation	ZKP Module	Generate random proof values	System should fail to verify the proof	
Brute Force OTP Attempt Simulation	MFA Module	Attempt multiple OTP guesses in a short time	Trigger rate-limiting or account lockout	
Unauthorized Certificate Verification Attempt	Verifier Access	Try to access certificate details without valid privileges	Access should be denied and attempt logged	

Non-Functional System Testing Procedures

To ensure the system's reliability, compatibility, and usability, various forms of testing will be performed. These include compatibility testing across platforms, performance benchmarking, accessibility evaluation for inclusive use, and user-friendliness testing with real users from target roles such as students, administrators, and HR personnel. Below is a summary of the testing procedures and expected results.

Table 12. *Non-Functional System Testing Procedures*

Test Type	Test Steps	Expected Outcome
Compatibility Testing	1. Open the system in different browsers: Chrome, Firefox, Edge, Safari. 2. Access all major pages (login, request, verification). 3. Repeat the test on Android devices with versions 9, 10, 11+.	UI renders consistently, no functional or visual errors across platforms.

Table 12 cont'd. *Non-Functional System Testing Procedures*

Performance Testing	1. Simulate multiple users logging in concurrently. 2. Perform multiple certificate requests and QR code generation. 3. Monitor server CPU, memory, and response time using tools like JMeter or Flask profiling.	System remains stable, responds within acceptable load time, no crashes or slowdowns.
Accessibility Testing	1. Enable screen reader (e.g., NVDA) and navigate system. 2. Attempt keyboard-only navigation. 3. Use Chrome Lighthouse to audit accessibility. 4. Check contrast ratios and alt texts.	Users with impairments can navigate and understand system functions effectively.
User-Friendliness Testing	1. Select participants: students, registrar staff, and HR staff. 2. Ask each user to perform role-based tasks (request, approve, verify certificate). 3. Collect feedback via a post-test form.	Users' complete tasks without confusion and provide positive feedback on usability.

User Acceptance Testing

User Acceptance Testing will serve as the final validation phase before full deployment of the Certificate Authentication System. This phase is important in assessing whether the system meets the expectations and functional requirements of actual end users, including registrars, graduates, and employers.

Objectives

- To verify that the system satisfies the functional and business requirements from the perspective of intended users.
- To ensure that users can perform real-world tasks such as certificate verification, issuance, and validation with minimal difficulty or error.
- To confirm that the cryptographic and QR-based verification features work reliably in practical workflows.

Test Participants

Table 13. *Participants for System Testing*

User Role	Number of Participants	Purpose
Registrar/Admin Staff	5	Certificate Issuance / QR Code Generation
Graduating Students	5	Certificate Viewing / MFA experience
HR Personnels	5	QR scanning, verification, validation experience

UAT Environment

- A pre-deployment staging environment identical to the production system.
- Simulated data based on real use cases.
- Devices: desktops (Windows/macOS), Android smartphones.

Table 14. *Participants Tasks for System Testing*

Task	User Role	Expected Outcome
Student applies for a certificate via their account	Student	Application is submitted and visible in registrar dashboard
Registrar reviews and issues a digitally signed certificate	Registrar	Certificate includes ECC-encrypted QR code
Student receives the certificate and completes MFA authentication	Student	Certificate is accessed after successful MFA
Employer scans QR and verifies the authenticity without full data	HR Officer	ZKP module confirms validity without exposing full content
Attempt verification with tampered QR	HR Officer	System detects invalid QR payload
Reissue request by student due to incorrect details	Student, Registrar	System logs request and reissues updated certificate

Acceptance Criteria

- No critical errors during task execution.
- All roles can complete tasks without technical support.
- Verification and authentication modules perform accurately and securely.
- MFA and ECC decryption operate consistently.
- QR payload is validated with hash integrity.

Feedback Collection Methods

- Observation and screen recording (with consent).
- Post-task questionnaire (e.g., System Usability Scale).
- Verbal or written comments from users.

This testing phase will provide formal approval from end users, building confidence in the system’s readiness for institutional deployment. Feedback and minor issues discovered here may lead to the final round of improvements before launch.

Evaluation Metrics

To assess the effectiveness, reliability, and usability of the system, several key performance metrics will be used. These metrics are designed to measure the system’s success in real-world usage scenarios, to ensure it meets both technical and user expectations. The following metrics will be observed during system testing and user validation phases. Each test scenario will be performed by actual users based on defined roles (students, administrators, verifiers/HR officers), and outcomes will be recorded for further analysis.

Percentage Calculation Method:

To quantify success and failure rates across different metrics, percentage values will be used. The formula for calculating percentage success (or failure) is:

$$\text{Percentage} = \left(\frac{\text{Successful Attempts}}{\text{Total Attempts}}\right) \times 100$$

This computation will be applied for tests such as QR Code scanning, Multi-Factor Authentication, and Certificate Verification to assess overall effectiveness.

Table 15. *System Evaluation Metrics*

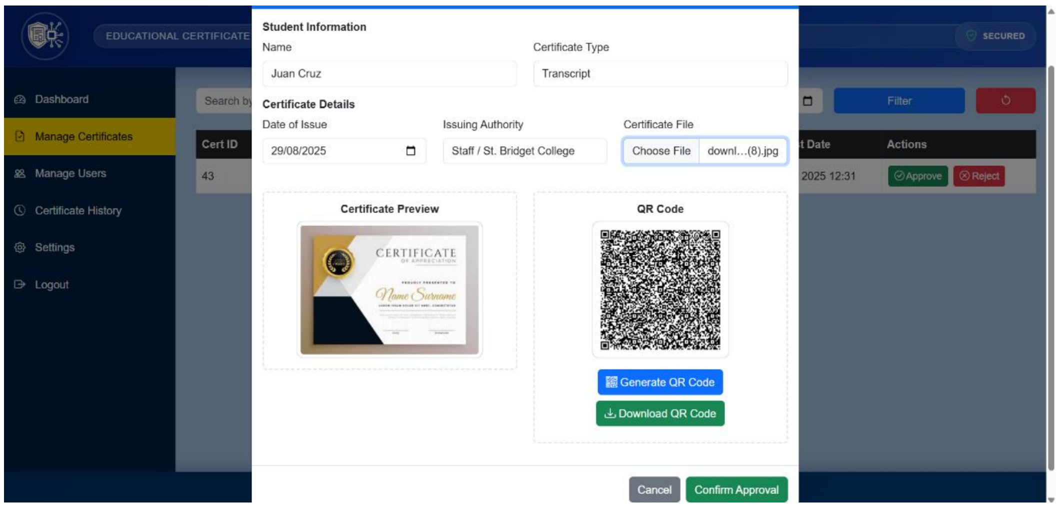
Metric	**Purpose**	**Test Scenario**	**Measurement Tool**	**Target Goal**
Certificate Verification Speed	Measures the average time to verify a certificate after QR scan and decryption.	Time how long it takes from QR scan to decrypted certificate display.	Stopwatch / Timestamp logs	≤ 3 seconds per certificate
Encryption Processing Time	Evaluates system performance in ECC encryption/decryption operations.	Measure how long it takes to encrypt and decrypt a certificate in the system.	System log timer / Python timing functions	≤ 2 seconds for both encryption & decryption
MFA Success/ Failure Rate	Determines the reliability of Multi-Factor Authentication during login.	Record the number of successful vs. failed MFA attempts during login trials.	System audit logs / MFA log reports	≥ 95% success rate
QR Code Scan Success Rate	Checks the reliability of QR code generation, scanning, and decoding.	Count how many QR scans correctly decode and verify certificate data.	QR scan logs / mobile device scanner feedback	≥ 98% success rate
ZKP Validation Accuracy	Assesses whether ZKP module correctly verifies certificate proof without full data.	Test simulated ZKP proofs and track if verification succeeds/fails as expected.	ZKP log analyzer / custom testing script	≥ 90% validation accuracy
System Accessibility	Measures ease of access across various user devices and roles.	Collect user feedback on device/browser access and basic usability.	Device/browser compatibility checklist	Full support for Chrome, Firefox, Android
User Satisfaction (SUS Score)	Evaluates overall user satisfaction with the system.	Use System Usability Scale (SUS) survey answered by user groups post-testing.	System Usability Scale (SUS) Questionnaire	SUS Score ≥ 80 (Good to Excellent)

CHAPTER 4

RESULTS AND DISCUSSION

System Implementation based on Research Objectives

The graphical user interfaces of the system were designed and developed in accordance with the objectives of the study. Each module's interface reflects its intended purpose, ensuring that the functionalities support secure, reliable, and user-friendly operations. The presented screenshots showcase how these objectives are translated into practical workflows and features within the system.



Student Information

Name: Juan Cruz

Certificate Type: Transcript

Certificate Details

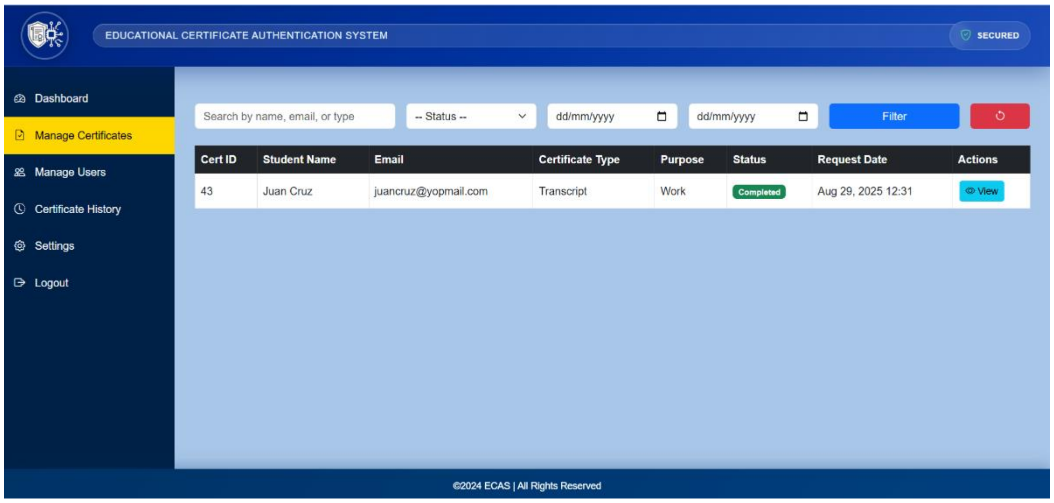
Date of Issue: 29/08/2025

Issuing Authority: Staff / St. Bridget College

Certificate File: Choose File downl...(8).jpg

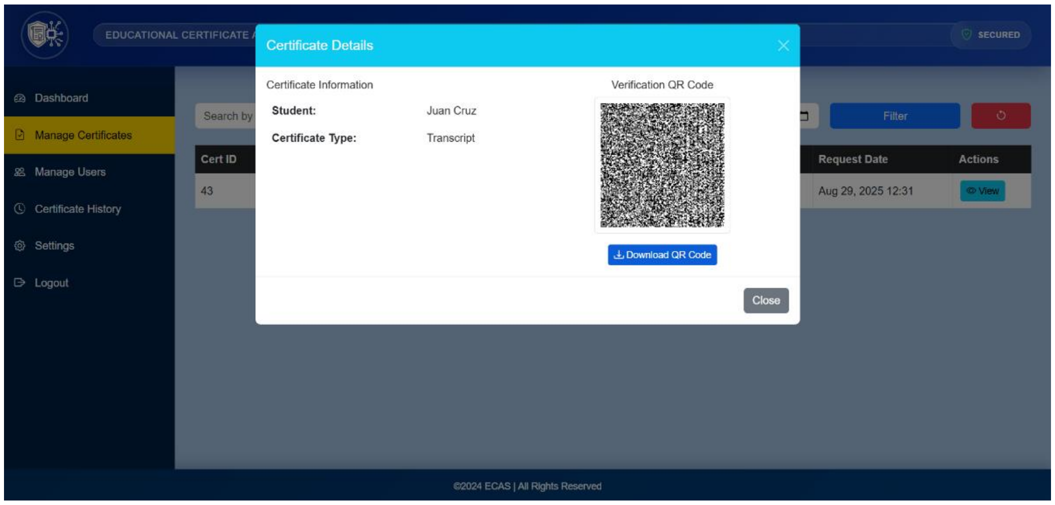
Cert ID	t Date	Actions
43	2025 12:31	Approve Reject

Figure 13. *Admin Module (Generate QR Code Page)*



Cert ID	Student Name	Email	Certificate Type	Purpose	Status	Request Date	Actions
43	Juan Cruz	juancruz@yopmail.com	Transcript	Work	Completed	Aug 29, 2025 12:31	View

Figure 14. *Admin Module (Manage Certificates Page)*



Certificate Information

Student:	Juan Cruz
Certificate Type:	Transcript

Figure 15. *Admin Module (Certificate Details Page)*

As shown in *Figure 13 (QR Code Generation Interface)*, *Figure 14 (Manage Certificates Page)*, and *Figure 15 (Certificate Details Page)*, the system demonstrates its ability to generate encrypted QR codes for each issued certificate. These QR codes encapsulate Zero-Knowledge Proof (ZKP) commitments and ECC-signed certificate data, which are processed securely in the backend. By embedding both cryptographic proofs and digital signatures, the generated QR codes ensure data integrity, authenticity, and resistance to tampering.

This implementation confirms that the system has successfully met Objective 1: To implement a QR code-based authentication module that securely encodes certificate data using strong encryption, digital signatures, and validation mechanisms.

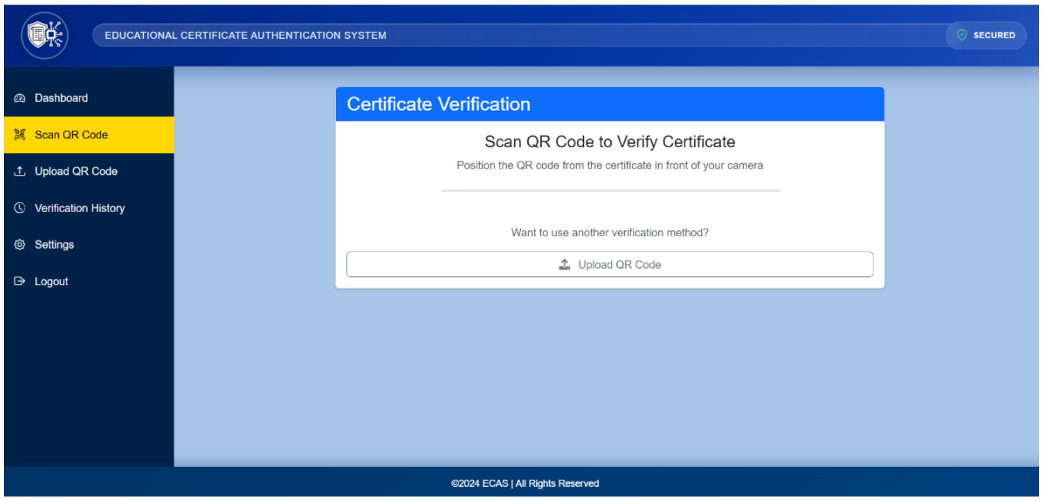

EDUCATIONAL CERTIFICATE AUTHENTICATION SYSTEM
SECURED
Dashboard
Scan QR Code
Upload QR Code
Verification History
Settings
Logout
Certificate Verification
Scan QR Code to Verify Certificate
Position the QR code from the certificate in front of your camera
Want to use another verification method?
Upload QR Code
©2024 ECAS | All Rights Reserved

Figure 16. *Verifier Module (Scan QR Code Page)*

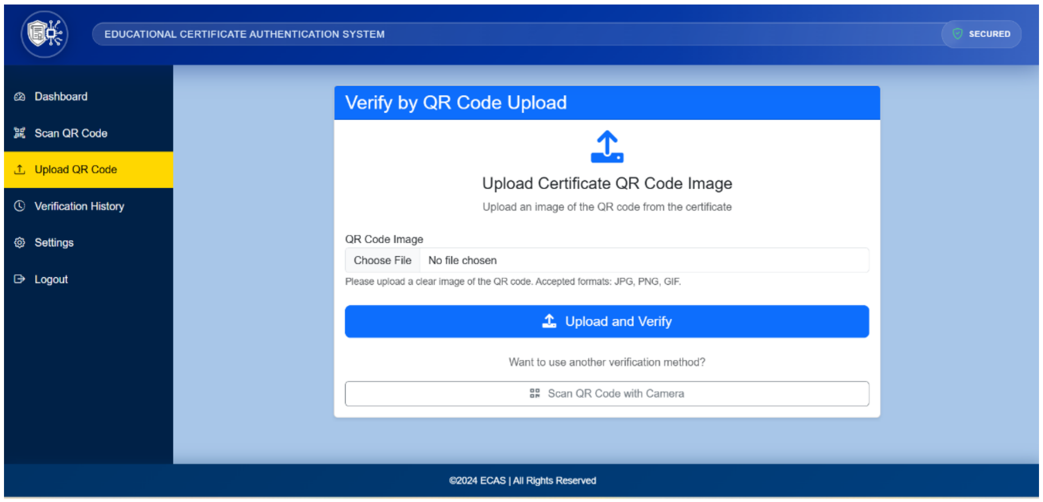
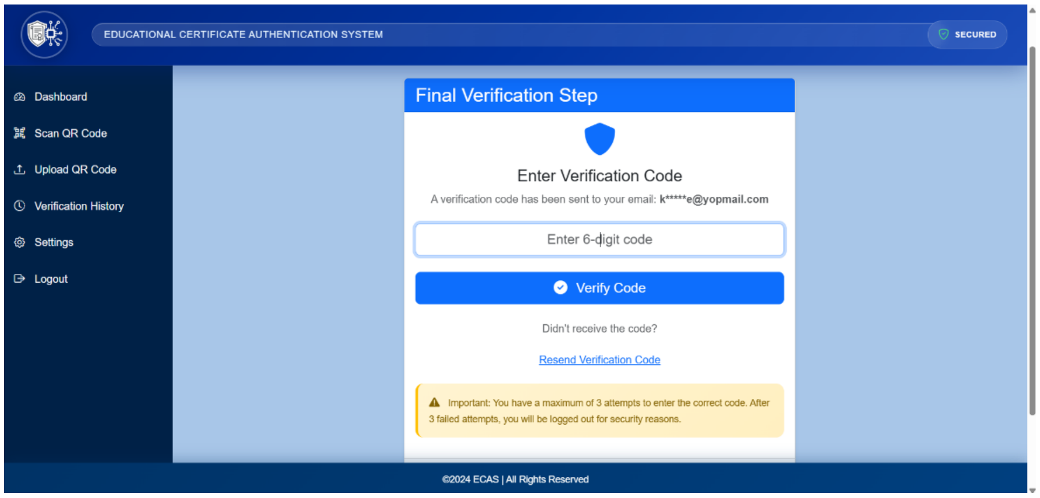

EDUCATIONAL CERTIFICATE AUTHENTICATION SYSTEM
SECURED
Dashboard
Scan QR Code
Upload QR Code
Verification History
Settings
Logout
Verify by QR Code Upload
Upload Certificate QR Code Image
Upload an image of the QR code from the certificate
QR Code Image
Choose File
No file chosen
Please upload a clear image of the QR code. Accepted formats: JPG, PNG, GIF.
Upload and Verify
Want to use another verification method?
Scan QR Code with Camera
©2024 ECAS | All Rights Reserved

Figure 17. *Verifier Module (Upload QR Code Page)*



Final Verification Step

Enter Verification Code

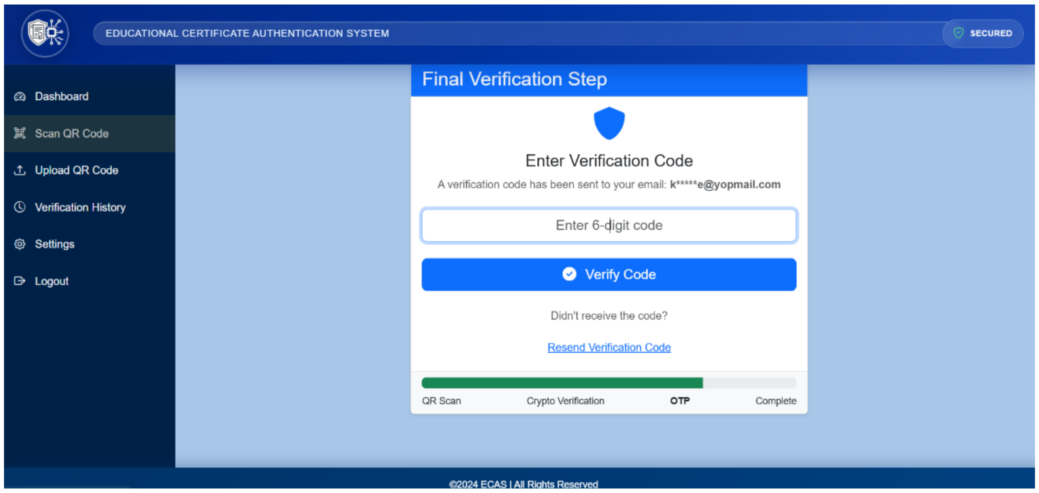
A verification code has been sent to your email: **k*****e@yopmail.com**

Didn't receive the code?

Important: You have a maximum of 3 attempts to enter the correct code. After 3 failed attempts, you will be logged out for security reasons.

©2024 ECAS | All Rights Reserved

Figure 18a. *Verifier Module (Verification Page)*



Final Verification Step

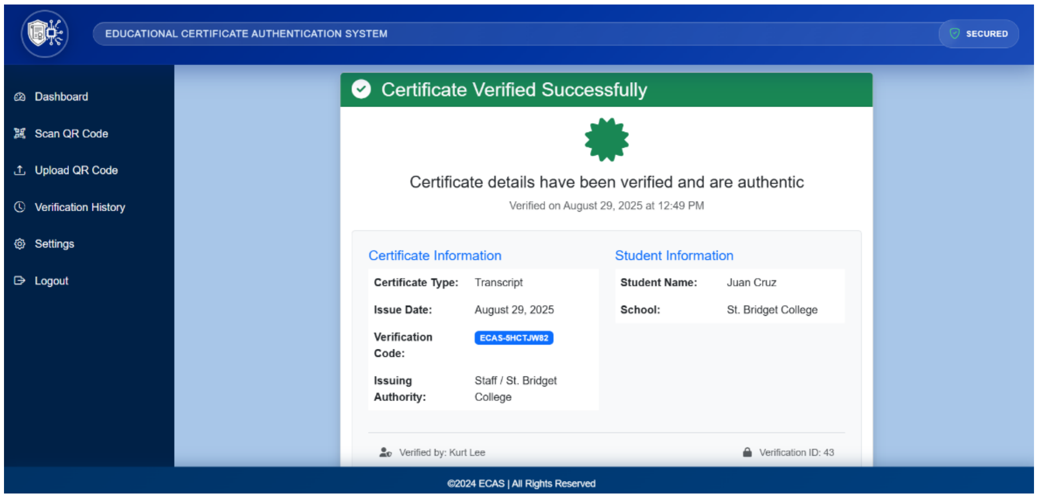
Enter Verification Code

A verification code has been sent to your email: **k*****e@yopmail.com**

Didn't receive the code?

©2024 ECAS | All Rights Reserved

Figure 18b. *Verifier Module (Verification Page)*



Certificate Verified Successfully

Certificate details have been verified and are authentic

Verified on August 29, 2025 at 12:49 PM

Certificate Information

Certificate Type:	Transcript
Issue Date:	August 29, 2025
Verification Code:	ECAS-5HCTJW82
Issuing Authority:	Staff / St. Bridget College

Student Information

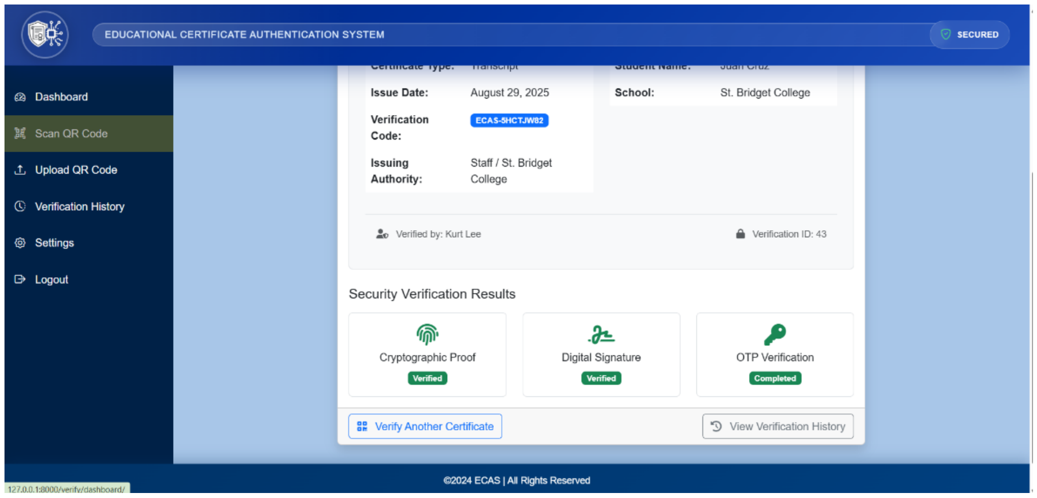
Student Name:	Juan Cruz
School:	St. Bridget College

Verified by: Kurt Lee

Verification ID: 43

©2024 ECAS | All Rights Reserved

Figure 18c. *Verifier Module (Verification Page)*


EDUCATIONAL CERTIFICATE AUTHENTICATION SYSTEM
SECURED
Dashboard
Scan QR Code
Upload QR Code
Verification History
Settings
Logout
Verify Another Certificate
View Verification History
©2024 ECAS | All Rights Reserved

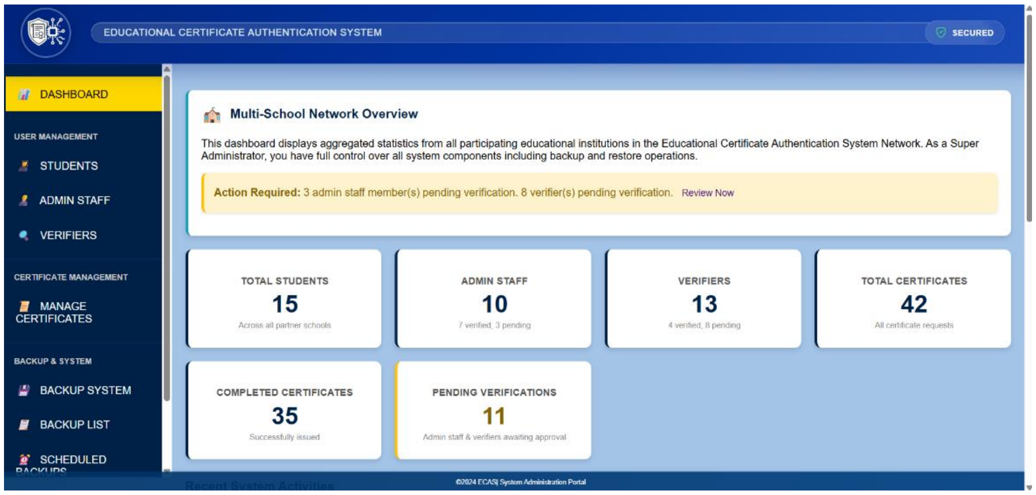
Certificate Type: Transcript
Student Name: Juan Cruz
Issue Date: August 29, 2025
School: St. Bridget College
Verification Code: ECAS-5HCTJW82
Issuing Authority: Staff / St. Bridget College
Verified by: Kurt Lee
Verification ID: 43

Security Verification Results

Cryptographic Proof Verified
Digital Signature Verified
OTP Verification Completed

Figure 18d. *Verifier Module (Verification Page)*

Screenshots in *Figure 16 (Verifier - Scan QR Page)* and *Figure 17 (Verifier - Upload Page)* illustrate how the verifier can initiate certificate validation either by scanning the QR code directly or by uploading the certificate file. *Figures 18a - 18d (Verifier Module - Verification Pages)* present the sequence of verification interfaces that demonstrate the system's ability to check certificate integrity and authenticity. At the backend, the certificate data is validated through ECC-based digital signatures and Zero-Knowledge Proof (ZKP) verification. In addition, Multi-Factor Authentication (MFA) is enforced through One-Time Password (OTP) validation before the final confirmation of certificate authenticity. This implementation demonstrates that the system successfully fulfills Objective 2: To enhance certificate authentication using Elliptic Curve Cryptography (ECC), Multi-Factor Authentication (MFA), and Zero-Knowledge Proof (ZKP).



Multi-School Network Overview

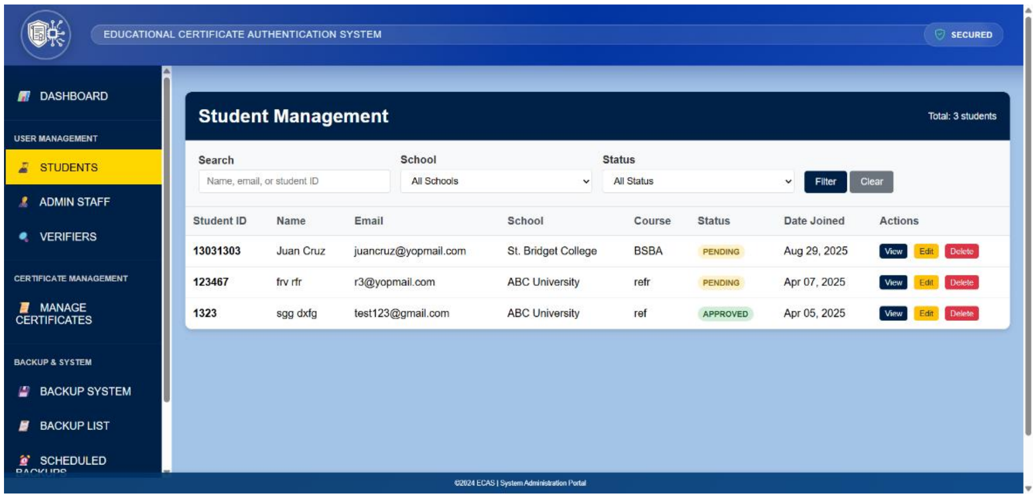
This dashboard displays aggregated statistics from all participating educational institutions in the Educational Certificate Authentication System Network. As a Super Administrator, you have full control over all system components including backup and restore operations.

Action Required: 3 admin staff member(s) pending verification. 8 verifier(s) pending verification.

TOTAL STUDENTS	15	Across all partner schools
ADMIN STAFF	10	7 verified, 3 pending
VERIFIERS	13	4 verified, 8 pending
TOTAL CERTIFICATES	42	All certificate requests
COMPLETED CERTIFICATES	35	Successfully issued
PENDING VERIFICATIONS	11	Admin staff & verifiers awaiting approval

Recent System Activities

Figure 19. *Super Admin Module (Dashboard)*

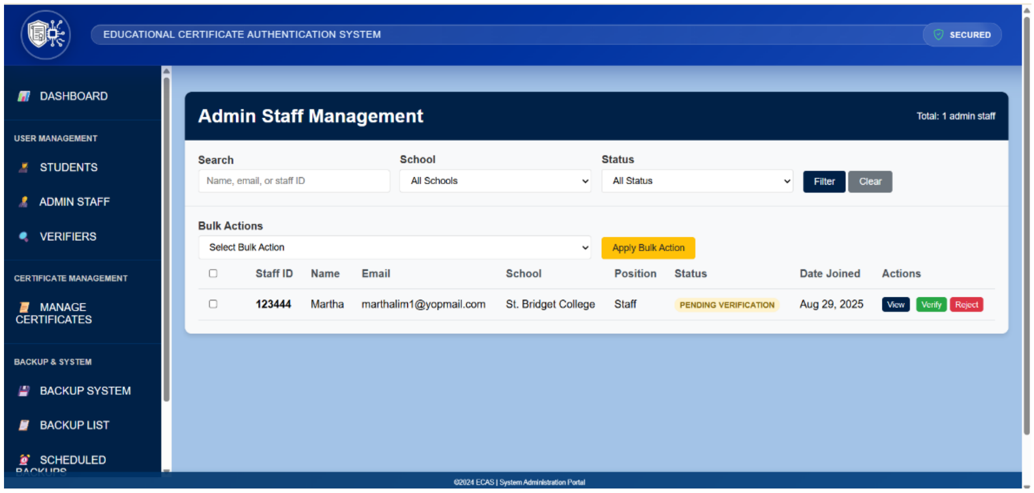


Student Management

Total: 3 students

Student ID	Name	Email	School	Course	Status	Date Joined	Actions
13031303	Juan Cruz	juancruz@yopmail.com	St. Bridget College	BSBA	PENDING	Aug 29, 2025	View Edit Delete
123467	frv rfr	r3@yopmail.com	ABC University	refr	PENDING	Apr 07, 2025	View Edit Delete
1323	sgg dxfg	test123@gmail.com	ABC University	ref	APPROVED	Apr 05, 2025	View Edit Delete

Figure 20. *Super Admin Module (Student Management Page)*

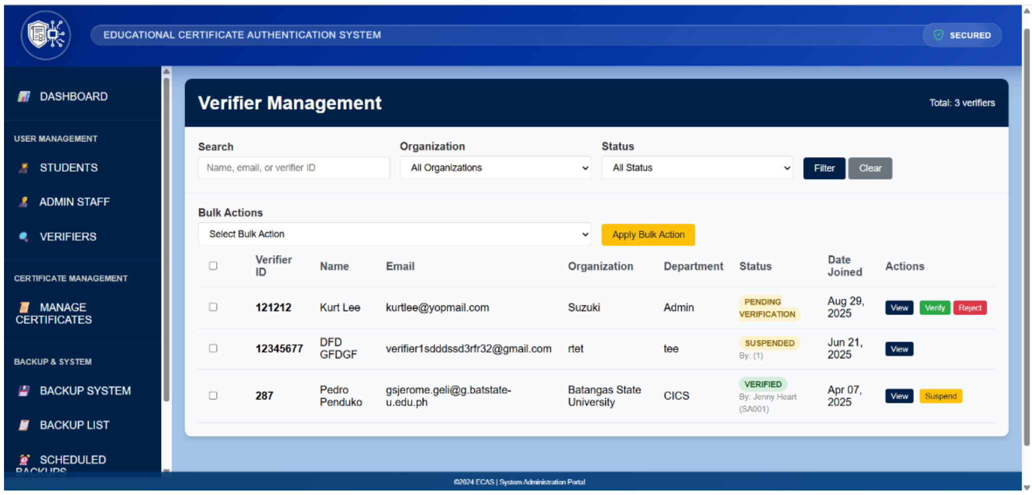


Admin Staff Management

Total: 1 admin staff

☐	Staff ID	Name	Email	School	Position	Status	Date Joined	Actions
☐	123444	Martha	marthalim1@yopmail.com	St. Bridget College	Staff	PENDING VERIFICATION	Aug 29, 2025	View Verify Reject

Figure 21. *Super Admin Module (Admin Staff Management Page)*



Verifier Management

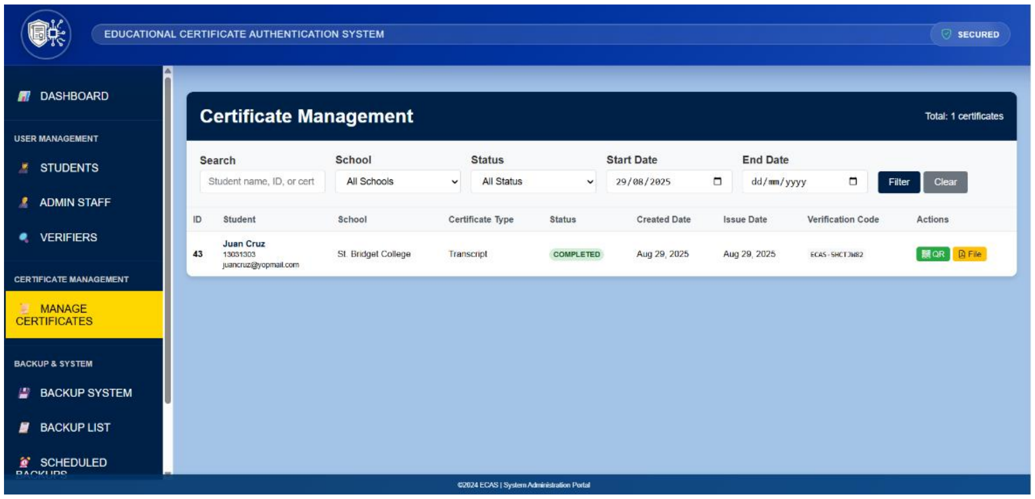
Total: 3 verifiers

Search: Name, email, or verifier ID
Organization: All Organizations
Status: All Status

Bulk Actions: Select Bulk Action

	Verifier ID	Name	Email	Organization	Department	Status	Date Joined	Actions
☐	121212	Kurt Lee	kurtlee@yopmail.com	Suzuki	Admin	PENDING VERIFICATION	Aug 29, 2025	View Verify Reject
☐	12345677	DFD GFDGF	verifier1sdddssd3rfr32@gmail.com	rtet	tee	SUSPENDED By: (1)	Jun 21, 2025	View
☐	287	Pedro Penduko	gsjerome.geli@g.batstate-u.edu.ph	Batangas State University	CICS	VERIFIED By: Jenny Heart (SA001)	Apr 07, 2025	View Suspend

Figure 22. *Super Admin Module (Verifier Management Page)*



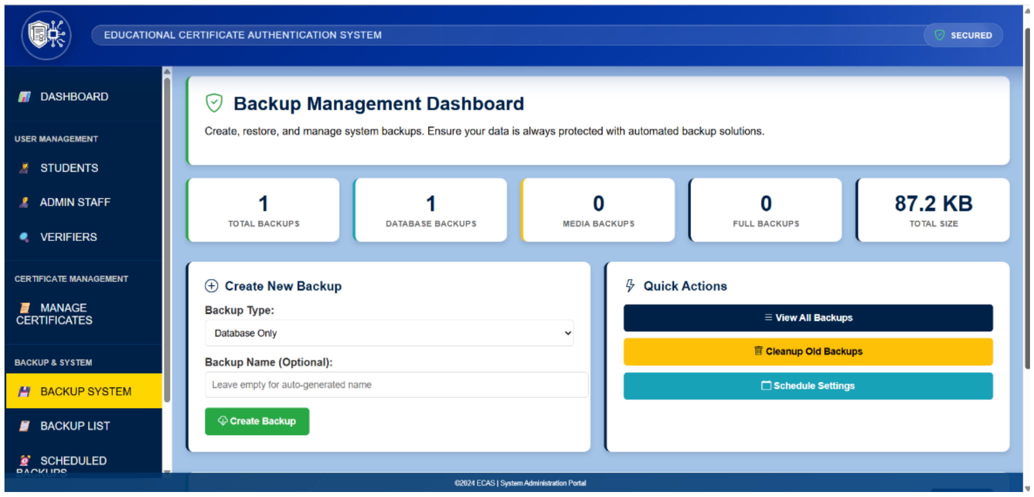
Certificate Management

Total: 1 certificates

Search: Student name, ID, or cert
School: All Schools
Status: All Status
Start Date: 29/08/2025
End Date: dd/mm/yyyy

ID	Student	School	Certificate Type	Status	Created Date	Issue Date	Verification Code	Actions
43	Juan Cruz 13031303 juancruz@yopmail.com	St. Bridget College	Transcript	COMPLETED	Aug 29, 2025	Aug 29, 2025	ECAS-SHCTJN82	QR File

Figure 23. *Super Admin Module (Certificate Management Page)*



Backup Management Dashboard

Create, restore, and manage system backups. Ensure your data is always protected with automated backup solutions.

1	1	0	0	87.2 KB
TOTAL BACKUPS	DATABASE BACKUPS	MEDIA BACKUPS	FULL BACKUPS	TOTAL SIZE

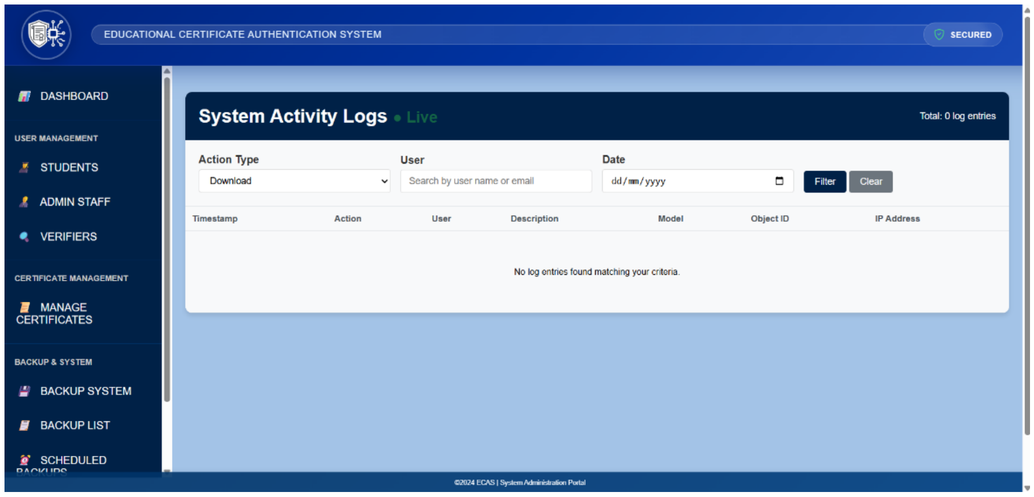
Create New Backup

Backup Type: Database Only

Backup Name (Optional): Leave empty for auto-generated name

Quick Actions

Figure 24. *Super Admin Module (Backup Management)*



System Activity Logs • Live

Total: 0 log entries

Action Type: Download

User: Search by user name or email

Date: dd/mm/yyyy

Timestamp	Action	User	Description	Model	Object ID	IP Address

No log entries found matching your criteria.

©2024 ECAS | System Administration Portal

Figure 25. *Super Admin Module (System Activity Logs)*

Screenshots in Figure 19 - 25 demonstrate how officers can oversee system operations, manage student and staff records, regulate verifier access, and handle certificate issuance with ECC signing, QR encoding, and ZKP integration. Backup management and activity logs further support maintenance and accountability. These functions confirm that the system fulfills Objective 3: To develop a certificate management system that enables secure issuance, user access control, monitoring, and record management.

Data Presentation and Analysis

The system was implemented based on the design and methodology outlined in Chapter 3. Each module was deployed and tested individually, then integrated into the full system. The implementation was guided by the objectives of the study, with features designed to meet the goals of secure certificate authentication, efficient request handling, and intuitive system access across all user roles.

Functional Testing Results

The testing and evaluation of the system produced results across different modules, including QR code generation and scanning, ECC-based encryption and digital signatures, Zero-Knowledge Proof (ZKP) integration, and Multi-Factor Authentication (MFA). Data were collected through system performance testing and user evaluation surveys.

Table 13 shows the unit and integration testing results for the core modules of the system. All major modules passed with correct outputs and no critical errors.

Table 16. *Unit and Integration Testing Results*

Test Case	Module	Expected Outcome	Actual Outcome	Results
QR Generation with Valid Certificate	QR Code Module	QR is successfully generated with correct encrypted data		
QR Parsing and Payload Extraction	QR Code Module	Parsed data matches the original input		
ECC Key Pair Generation	ECC Module	Keys are created correctly and stored securely		
ECC Encryption of Certificate	ECC Module	Ciphertext is not readable and decrypts correctly		
ECC Decryption of Encrypted Payload	ECC Module	Output matches original certificate		
MFA Code Generation	MFA Module	Code is generated and valid for time window		
MFA Code Verification	MFA Module	Verification succeeds or fails appropriately		
ZKP Proof Creation	ZKP Module	Proof values generated successfully		
ZKP Verification	ZKP Module	Verification passes for correct proof, fails otherwise		
Certificate Request Processing	Certificate Module	System stores request and triggers workflow		
Certificate Issuance	Admin/Registrar Module	Certificate generated and QR is attached		

Verify Certificate Using QR	QR + ECC + ZKP Modules	System verifies certificate without exposing full details		
User Login with MFA	Login + MFA Module	Access granted after both credentials and OTP are correct		
Admin/Staff User Role Functionality	Dashboard + DB Modules	Roles access only allowed functions		
Certificate Database Read/Write	Backend + Database			
Verifier Certificate Lookup	Verifier + Backend			

Table 14 shows the security testing results for the core modules of the system. All major modules passed with correct outputs and no critical errors.

Table 17. *Security Testing Results*

Test Case	**Module**	**Test Description**	**Expected Outcome**	**Actual Outcome**	**Results**
Invalid QR Payload Format	QR Code Module	Input a tampered or malformed QR code	System should reject the code and display an error message		
Expired QR Code	QR Code Module	Use a QR code beyond its validity period	System should deny verification and inform the user		
Mismatched ECC Public Key	ECC Module	Decrypt with incorrect public key	Decryption should fail or produce invalid content		
Simulated Hash Tampering	ECC/QR Module	Modify the hash value inside the QR payload	System should detect tampering and halt the process		
MFA OTP Input Failure	MFA Module	Input wrong OTP multiple times	Account should be temporarily locked or flagged		
OTP Replay Attack Simulation	MFA Module	Reuse a previously valid OTP	System should detect and deny the attempt		
Incomplete ZKP Proof Submission	ZKP Module	Submit partial proof values	System should reject the incomplete proof		
Invalid ZKP Proof Simulation	ZKP Module	Generate random proof values	System should fail to verify the proof		
Brute Force OTP Attempt Simulation	MFA Module	Attempt multiple OTP guesses in a short time	Trigger rate-limiting or account lockout		

Unauthorized Certificate Verification Attempt	Verifier Access	Try to access certificate details without valid privileges	Access should be denied and attempt logged		

Table 15 shows the other system test results for the core modules of the system. All major modules passed with correct outputs and no critical errors.

Test Type	**Test Steps**	**Expected Outcome**	**Actual Outcome**	**Results**
Compatibility Testing	1. Open the system in different browsers: Chrome, Firefox, Edge, Safari. 2. Access all major pages (login, request, verification). 3. Repeat the test on Android devices with versions 9, 10, 11+.	UI renders consistently, no functional or visual errors across platforms.		
Performance Testing	1. Simulate multiple users logging in concurrently. 2. Perform multiple certificate requests and QR code generation. 3. Monitor server CPU, memory, and response time using tools like JMeter or Flask profiling.	System remains stable, responds within acceptable load time, no crashes or slowdowns.		
Accessibility Testing	1. Enable screen reader (e.g., NVDA) and navigate system. 2. Attempt keyboard-only navigation. 3. Use Chrome Lighthouse to audit accessibility. 4. Check contrast ratios and alt texts.	Users with impairments can navigate and understand system functions effectively.		
User-Friendliness Testing	1. Select participants: students, registrar staff, and HR staff. 2. Ask each user to perform role-based tasks (request, approve, verify certificate). 3. Collect feedback via a post-test form.	Users' complete tasks without confusion and provide positive feedback on usability.		

System Performance Metrics

User Evaluation Results

Comparative Discussion

Discussion of Findings

CHAPTER 5

SUMMARY, CONCLUSIONS, AND RECOMMENDATIONS

Summary of Findings

Implications of Findings

Future Research Directions

Bibliography

Das, D., Abu Bakar, M. I., Rajshahi University, Faculty of Engineering, & Institute of Electrical and Electronics Engineers. (2019). *5th International Conference on Computer, Communication, Chemical, Materials and Electronic Engineering (IC4ME2): 11-12 July 2019*. Bangladesh Section.

Omerasevic, D., Behlilovic, N., & Mrdovic, S. (2014). An implementation of secure key exchange by using QR codes. *Proceedings ELMAR-2014*, 1-4. https://doi.org/10.1109/ELMAR.2014.6923341

Pangan, A. M. S., Lacuesta, I. L., Mabborang, R. C., & Ferrer, F. P. (2022). Authenticating data transfer using RSA-generated QR codes. *European Journal of Information Technologies and Computer Science, 2*(4), 18–30. https://doi.org/10.24018/compute.2022.2.4.73

El-Taj, H. R., & Alhadhrami, R. (2020). CryptoQR system based on RSA. *International Journal of Computer Science and Information Security, 18*(6).

Irawan, A. I., Santoso, I. H., Istikmal, & Rahayu, M. (2024). Implementation of QR code attendance security system using RSA and hash algorithms. *Jurnal Nasional Teknik Elektro dan Teknologi Informasi, 13*(1), 53-59. https://doi.org/10.22146/jnteti.v13i1.4395

Wibiyanto, A., & Afrianto, I. (2018, August). QR code and transport layer security for licensing documents verification. In *IOP Conference Series: Materials Science and Engineering* (Vol. 407, No. 1, p. 012069). IOP Publishing.

Naresh, K., & Pillai, P. N. (2014). QR verification system using RSA algorithm. *International Journal of Innovation and Scientific Research, 10*(2), 433-437.
Yasa, K. A., Sukarata, P. G., Putra, I., Nugroho, I., & Astawa, I. (2021). Secure electronic document with QR code and RSA digital signature algorithm.

Durafe, A. (2020). Securing criminal records using R-Pi, QR code and steganography. *International Journal of Innovative Technology and Exploring Engineering (IJITEE), 9*(6), 3075-3081.

Gupta, P., Saini, S., & Lata, K. (2017). Securing QR codes by RSA on FPGA. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)* (pp. 2289-2295). Udupi, India. https://doi.org/10.1109/ICACCI.2017.8126188

Yuan, B., Zhao, X., & Cui, D. (2018, January). The effective QR code technology development. In *2017 4th International Conference on Machinery, Materials and Computer (MACMC 2017)* (pp. 433-436). Atlantis Press.

Krinz, C. (n.d.). *Digital signatures: Principles, protocols, and applications*. Retrieved July 31, 2024, from https://sites.cs.ucsb.edu/~ckrintz/papers/security/crypto-tutorial

Stallings, W. (n.d.). *Cryptography and network security: Principles and practice*. Retrieved July 31, 2024, from https://dl.hiva-network.com/Library/security/Cryptography-and-network-security-principles-and-practice.pdf

QR Code Generator. (n.d.). *QR Code Generator*. QR Code Generator. Retrieved July 31, 2024, from https://www.qr-code-generator.com/

Denso Wave Incorporated. (n.d.). *QR Code.com*. Denso Wave Incorporated. Retrieved July 31, 2024, from https://www.qrcode.com/en/

Barcode FAQ. (n.d.). *QR Code basics*. Barcode FAQ. Retrieved July 31, 2024, from https://www.barcodefaq.com/qr-code/

National Privacy Commission. (2022). *In re: University of the Philippines Visayas resolution*. Retrieved from https://privacy.gov.ph/wp-content/uploads/2024/04/NPC-BN-18-045-2022.11.10-In-re-University-of-the-Philippines-Visayas-Resolution-Final.pdf

Philippine Star. (2020, January 16). Singapore jails Filipina for faking diploma, records. *The Philippine Star*. https://www.philstar.com/headlines/2020/01/16/1985311/singapore-jails-filipina-faking-diploma-records

Encarnacion, A. (2022, July 31). Filipinos advised to be careful of new QR code scam. *NoypiGeeks*. https://www.noypigeeks.com/spotlight/qr-code-scam-quishing/

SSL2BUY. *RSA 4096-bit key in code signing certificate*. SSL2BUY. Retrieved December 6, 2024, from https://www.ssl2buy.com/wiki/rsa-4096-key-in-code-signing-certificate

Johannes. (2021, December 30). *Certificate length: 2048 or 4096 bit keys?* Semaphor. Retrieved December 6, 2024, from https://blog.semaphor.dk/20211230T1728

https://cheapsslsecurity.com/p/ecc-vs-rsa-comparing-ssl-tls-algorithms/
https://psa.gov.ph/content/advisory

https://seekpass.co/ph/credentials/digital-identity?internal_link=true&utm
https://psa.gov.ph/content/psa-introduces-secpa-e-verification-mobile-app
https://everify.gov.ph/

https://medium.com/asecuritysite-when-bob-met-alice/how-do-you-encrypt-with-elliptic-curves-20a35ccffe0a

https://www.researchgate.net/figure/Digital-signature-algorithm-III-ELLIPTIC-CURVE-CRYPTOGRAPHY-ECC_fig1_261447093

https://www.dock.io/post/zero-knowledge-proofs#how-zero-knowledge-proof-works
https://github.com/nakov/Practical-Cryptography-for-Developers-Book/blob/master/asymmetric-key-ciphers/ecc-encryption-decryption.md
https://www.includehelp.com/cryptography/digital-signature-algorithm-dsa.aspx