Hash-Based Authentication Code Algorithm for Quick Response (QR) Code as Digital Signature

*Nur Khairiyah[1], Yeka Hendriyani[1], Ahmaddul Hadi[1], Lativa Mursyida[1]

[1] *Department,of Electronics Engineering, Padang, Indonesia*

**Corresponding Author: nkhairiyah018@gmail.com*

***Abstract* - The authenticity and integrity of digital documents are crucial in modern data exchange. To ensure these qualities, cryptographic methods like digital signatures are employed. This research aims to develop a document authentication system using the hash-based authentication code algorithm, combined with QR codes and digital signatures. The system is designed to verify the authenticity of internship certificates issued by PT. ARG Solusi Technology. The digital signature process involves generating a message digest using the Secure Hash Algorithm-256 (SHA-256) and encrypting it with the Hash-Based Authentication Code (HMAC) algorithm. This digest is then embedded in a QR code. The system is developed using the waterfall methodology, which includes requirements analysis, system design, implementation, testing, deployment, and maintenance phases. It is implemented as a web service using RESTful API and tested using white-box testing, where the main functions are evaluated for their operation. The results indicate that the system effectively ensures the authenticity and integrity of documents, preventing document forgery.**

***Keywords*— *Hash-Based Authentication Code*, Digital *Signature*, *RESTful API, Web Service*.**

I. INTRODUCTION

Globalization has driven an era of information technology development, leading to the emergence of criminal activities [1]. It cannot be denied that certain individuals misuse technological advancements for personal or group gains, thereby violating legal regulations. One such criminal activity in the world of technology is known as Cyber Terrorism. Cyber Terrorism entails acts of terror against computer systems, networks, and stored information within computers [2]. One form of such actions falling under Cyber Terrorism is Cyber sabotage and extortion. Cyber sabotage and extortion constitute criminal acts involving disruption, damage, or destruction of data, computer programs, or internet-connected network systems [3]. One of the cyber sabotage and extortion crimes includes electronic document forgery [4].

DOI: 10.24036/int.j.emerg.technol.eng.educ..v1i1.2

Corresponding author: Nur Khairiyah
Universitas Negeri Padang
Email: nkhairiyah018@gmail.com

Received: 05-08-2024
Revised: 13-08-2024
Accepted: 04-10-2024
Published:06-10-2024

Documents are important assets for agencies, organizations, states, or individuals, and data loss can pose a risk if not properly safeguarded. With the advancement of information technology, document forgery has become easier, making authentication crucial to maintaining document integrity [5]. One form of effort in protecting document integrity is through signatures [6].

Digital signature is an authentication system that allows message senders to add a code as a representation of their signature [7]. Digital signatures are highly crucial in digital documents, enhancing operational efficiency and ensuring compliance with data security regulations [8]. They provide assurance that documents are authentic and unaltered, reducing the risks of fraud and legal disputes. Therefore, digital signatures become a pivotal aspect in digital transformation and information security.

Document forgery is often carried out by manipulating the contents to create copies that closely resemble the originals, thus increasing the need to maintain the confidentiality and authenticity of documents. In the current technological development, manual verification becomes less effective as it requires time and complex procedures.

PT. ARG Technology Solutions, a technology information company in Padang City, West Sumatra, has been in operation since 2022 and has been offering internship opportunities to vocational high school students since 2023. More than 50 students have been accepted into this internship program, providing them with practical experience and official certificates as recognition of their participation. Although Argenesia has strict policies regarding the issuance and verification of internship certificates, the process of signing and validating certifications is still carried out manually by the company's CEO. Therefore, this research selects PT. ARG Technology Solutions as the case study location.

In this study, a technique is developed to create a digital signature generation system aimed at providing authentication for digital certificates to verify the authenticity of certificates at PT. ARG Solusi Teknolosi. This technique utilizes one of the cryptographic techniques known as the Hash-Based Authentication Code Algorithm.

Hash-based Message Authentication Code (HMAC) is a cryptographic authentication technique that utilizes a hash function and a secret key [9]. In HMAC, the message or data to be authenticated is processed using a hash function along with a secret key known only to the authorized parties [10]. The result of this hash operation is then used to generate an authentication code called HMAC. This algorithm provides a higher level of security by offering a robust authentication mechanism, while leveraging the concept of digital signatures [11].

By implementing the HMAC algorithm to perform hashing processes that include data, payload, and document signatures to be embedded in a QR Code. QR (Quick Response) code is a type of two-dimensional matrix barcode that stores information and can be read using a camera [12]. The QR code consists of black and white squares in a rectangular grid, storing binary data read by QR code readers .

This digital signature is expected to maintain certificate authentication and reduce the risk of crimes such as data fraud. Furthermore, a system will be developed to ensure that the QR Code can only be read and verified by authorized parties, thus enhancing document security and integrity more effectively. The system features key functionalities including generating a hashed message digest and printing it in QR Code format, as well as printing documents and document verification. The system will be built on a web service-based architecture using RESTful API.

Web services and RESTful APIs enable interaction between various applications and systems, including in system implementations [13]. Web services serve as connectors to generate, hash, and verify QR codes as well as related documents. RESTful APIs allow applications or users to communicate [14] with the server for message digest creation, QR code generation, document printing, and document verification.

Therefore, this study aims to implement the Hash Based Authentication Code Algorithm as a digital signature process on internship certificates at Argenesia to study the signature generation process and verification and validation processes based on Quick Response Code. This research will utilize Spring Boot as the backend framework.

II. RESEARCH METHOD

The design in this study used the Software Development Life Cycle (SDLC) method with the waterfall process model [15]. The waterfall method is one of the most straightforward system development methods [16]. In this chapter, the author follows the steps of the waterfall method, which consist of Requirements Gathering and Analysis, Design, Systems Implementation and Coding, Testing, and Maintenance. The process in this research is illustrated in the following diagram.

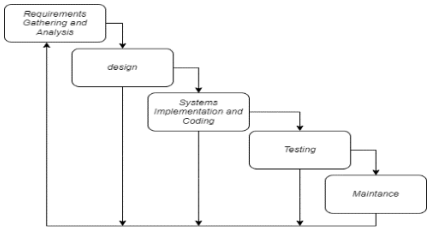

Fig 1. Waterfall Method

The requirements gathering and analysis stage involves collecting and thoroughly evaluating all the application's needs. Based on these requirements, the design phase develops the application's components, workflow, and operational processes. In this research, Canva and Draw.io were used for designing.

During the system implementation and coding phase, the focus shifts to translating the system design into functional software. This involves developing code according to the specifications and ensuring seamless integration between system components. Java, with the Spring Boot framework, was the programming language chosen for this study.

The testing phase ensures that the system operates flawlessly and meets the established objectives. White-box testing, utilizing unit testing with JUnit, was employed to validate the system's functionality.

III. RESULT AND DISCUSSION

A. Requirements Gathering and Analysis

The research will conduct an analysis of the current business processes at PT. ARG Solusi Teknologi and assess the necessary system functionalities to resolve any arising issues. This analysis is derived from information and data gathered through interviews and observations with the CEO of PT. ARG Solusi Teknologi.

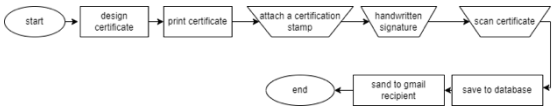

Fig 2. Proposed system analyst for signature generating and signature verification

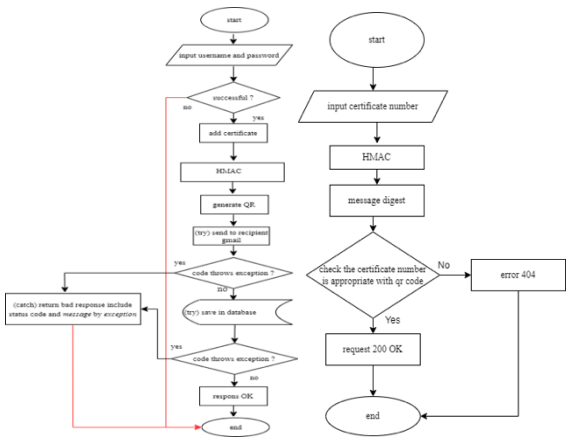

Fig 3. Proposed system analyst for signature generating and signature verification

Based on observations and interviews, PT. ARG Solusi Teknologi signs internship certificates manually and affixes a stamp as evidence of the certificate's validity and authenticity.

This research proposes the implementation of the Hash-Based Authentication Code algorithm to provide digital document authentication for verifying the authenticity of web service-based certificates. The system includes a digital signature generator in the form of a QR code and verification feature to validate the integrity of certificates.

B. Design

After analysis, the research progresses into the design phase based on the findings from the requirement gathering and analysis stage [17]. The research employs UML diagrams (Unified Modeling Language) such as activity diagrams, sequence diagrams, and class diagrams.

1. Use Case Diagram

   The use case diagram illustrates the functionality of the QR Code generator system and certification validation using the Hash-Based Authentication Code algorithm. The administrator is responsible for creating, sending, and storing certificates in the database, while relevant institutions or other parties can validate the letter certificates. Below are the designed use cases depicted in the diagram.

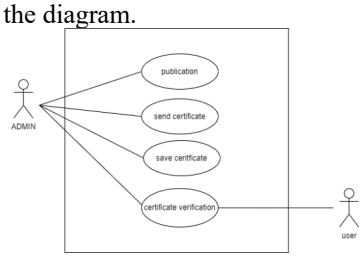

Fig 4. Use Case Diagram

2. Sequence Diagram

   The Sequence Diagram illustrates the interaction among objects within the system and its environment through messages along the time axis. Its purpose is to facilitate and guide system design [18]. The sequence diagram designed can be viewed in the image below.

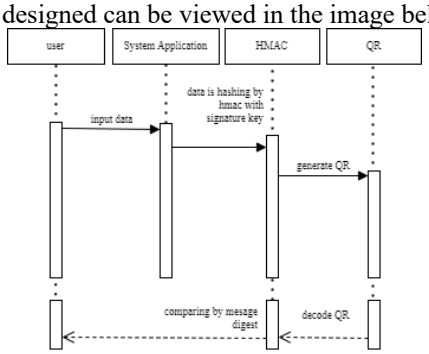

Fig 5. Sequence Diagram

3. Class Diagram

   A class diagram is a visual representation of the class structure within a system, displaying the attributes and methods owned by each class as well as the relationships between classes. This diagram aids in modeling and visualizing the static structure of the system clearly and in detail [19]. The following class diagram, designed for reference, can be seen in the image below.

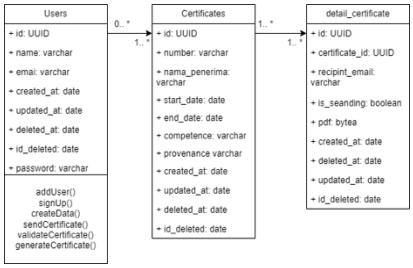

Fig 6. Class Diagram

C. Systems Implementation and Coding

Federal Information Processing Standards (FIPS) Publication is a series of standards issued by NIST to regulate various aspects of information processing and computer security within the US government, including cryptography, key management, and data protection. Standards for verification and validation in digital signature algorithms are outlined in FIPS PUB 186-4. These standards will serve as references in constructing digital signatures and validating signatures using hash-based authentication code algorithms.

The implementation of the Hash Based Authentication Code algorithm as a digital signature in this research begins with inputting the data to be signed into the database. The data will be retrieved from the database in the form of objects. These objects contain an ID, certificate number, certificate owner's name, competencies, student's origin, internship start date, internship end date, and certificate issuance date. These objects will serve as the input value for the hash function. The function works to produce a message digest from the hashing result using the hash based authentication code algorithm. The message digest is generated from the hashing process of data with a 256-bit private key. The cryptographic hash function used is SHA-256. Below is the determination of the signature key and the creation of the Message Authentication Code (MAC) object with SHA-256, as well as the initialization of the object for use. The message digest obtained from the previous hashing process will be inserted into the quick response code on the certificate. The digital signature and QR code generated by the system in this study can be viewed in the following table.

TABLE I
REQUEST AND RESPONSE DATA

Input Data	Nama : Nur Khairiyah Id : ddfe6d7a-67ef-4cba-b63f-2b26b5a326a3 Kompetensi : Programmer Asal Sekolah : Universitas Negeri Padang Tanggal mulai : 17 April 2023 Tanggal akhir : 17 Mei 2023 Tanggal sertifikat : 17 Mei 2024
Private Key	6dbf751179df9f302f8956ddc 3a1af50d71ea1fc22fe06d1cc7674a2019e56e9
Message Diggest	tWDULgnObZd659IZEzW7G6 /pJWLfje2krDDs5cYSWnQ=
QR Code	

The Message Digest obtained from the hashing process using the Hash-Based Authentication Code (HMAC) algorithm is generated by combining data found in the database with a secret key initialized in the previous stage. The following are the steps on how HMAC generates the Message Digest:

TABLE III
DATA PREPARATION

Message :	“LetterSignatureDTO(id= f6ad1e2c-ac50-46e7-93f2-1ee32adb6723, sender=Nur Khairiyah, number= 0118/SM/ARG/2023, content=null, subject=null, place=null, recipient=null, city=null, source=null, signatory=null)”
Key :	6dbf751179df9f302f8956ddc3a1af50d71ea1fc22fe06d1cc7674a2019e56e9
Ipda SHA-256	x36363636 36363636 36363636 36363636 36363636 36363636 36363636 36363636 36363636 36363636 36363636 36363636 36363636 36363636 36363636 36363636 (x36 diulang 64 kali)

1. If the key length is less than the block size:
   The secret key length is 48 bytes, while the block size for SHA-256 is 64 bytes. Therefore, pad the key with zeros to make it 64 bytes.

   K0 =
   6dbf751179df9f302f8956ddc3a1af50d71ea1fc22fe06d
   1cc7674a2019e56e9000000000000000000000000000
   000000000000000000000000000000000

2. Perform K0 XOR ipad (0x36 repeated 64 times):

   ipad (0x36 repeated 64 times) =
   3636363636363636363636363636363636
   3636363636363636363636363636363636
   36363636363636363636363636363636

3. Concatenate the result with the message:

   (K0 XOR ipad) || message =
   5bed43374eefe6421dc26cbf061b9923e12c8bcf14d936
   c39c51619e379778d73636363636363636363636363636
   363636363636363636363636363636 ||
   LetterSignatureDTO(id=f6ad1e2c-ac50-46e7-93f2-1ee32adb6723, sender=Nur Khairiyah, number=0118/SM/ARG/2023, content=null, subject=null, place=null, recipient=null, city=null, source=null, signatory=null)

4. Hash the result with SHA-256:

   sha256((K0 XOR ipad) || message) =
   6ac27916a047874f1b4722df0e85f15e7c5dc3dbb7b59e
   9ad4e5be202ff78984

5. Perform K0 XOR opad (0x5C repeated 64 times):

   opad (0x5C repeated 64 times) =
   5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c5
   c5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c
   5c5c5c5c5c5c5c5c5c5c5c5c5c5c

6. Concatenate the result with the hash from step 4:

   (K0 XOR opad) || sha256((K0 XOR ipad) || message) =
   313b291d2575bb1803e912817956fb0c8344d3a5ab96a
   9dfffc52ca42bfc0ba15c5c5c5c5c5c5c5c5c5c5c5c5c
   5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c5
   c5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c ||
   6ac27916a047874f1b4722df0e85f15e7c5dc3dbb7b59e
   9ad4e5be202ff78984

7. Hash the result with SHA-256:

   HMAC = sha256((K0 XOR opad) || sha256((K0 XOR ipad) || message)) =
   9ab3ed0dbdbd37b7b9cf67224fd362372c6cfe9b2e230d
   6161ad13e9e4f6f3cb

8. Final HMAC for the message with the given secret key:

   9ab3ed0dbdbd37b7b9cf67224fd362372c6cfe9b2e230
   d6161ad13e9e4f6f3cb

The certificate verification system in this research necessitates the Quick Response Code decoding process. Decoding is the process of deciphering the code to extract the information stored within the QR Code. An interface has been constructed for facilitating the verification of certificates by the relevant parties. The initial certificate verification page functions to input necessary data, such as entering the certificate number and scanning the QR code.

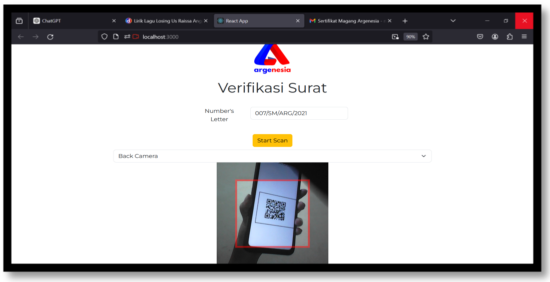

Fig 7. Input Certificate Number and Scan QR Code Page

Upon successful verification of the certificate, the application will present various details pertaining to the certificate, including the Certificate ID stored in the database, the date of issuance, the certificate number, and the email address of the recipient. The following is the interface displayed upon successful verification.

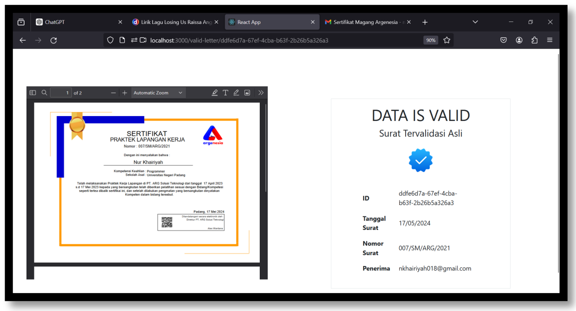

Fig 8. Verfication Certificate Page

Should a certificate be unable to undergo verification due to an erroneously entered certificate number and if the certificate remain unverified due to the presence of a counterfeit digital signature, the interface will be displayed as depicted in the image below.

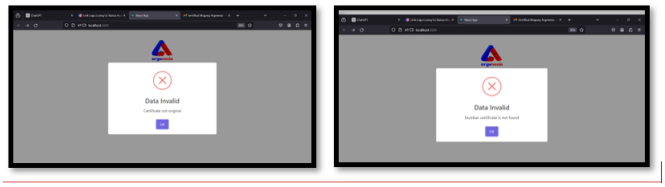

Fig 9. Invalid Verfication because signature is false and number certificate is not found

D. Testing

This study utilizes white-box testing through unit tests to evaluate the system's programming code. Three specific unit tests were developed: HmacTest is designed to assess the functionality of the hash-based message authentication code algorithm to ensure the program performs as intended; LoginTest is intended to verify the programming code of the login feature to confirm it operates correctly; and VerificationTest aims to examine the program logic for signature verification to guarantee it works as expected.

1. Hash-Based Authentication Code Algorithms Unit Testing

   This test aims to verify the logic of the login feature's code. Initially, the setup function prepares the login data, which includes a random UUID, email, and password. This data is then passed to the testSignIn function, which simulates the authentication system's response and checks its accuracy. Unit testing is conducted to ensure the verification scenarios are executed correctly. Below is the scenario layout for the unit tests.

TABLE IIIII
TEST CASE FOR GENERATE MESSAGE DIGGEST

Number	1.1
Purpose	Setup Test Environment
Annotation	@BeforeEach
Method	testGenerateHMAC()
Steps	To set up the test environment, create a fake Certificate object. Mock the behavior of the getSender method on Certificate to return a fake User object, and mock the getName method on User to return the sender's name. Configure the mock certificateRepository to return the Certificate object when the findById method is called with a valid certificate ID. Then, call the generateHMAC method from qrService with the certificate ID. Verify that the generated HMAC is not null.
Results	passed
Number	1.2
Purpose	Execute Test for generateHMAC with Invalid Certificate ID
Annotation	@Test
Method	testGenerateHMACwithInValid()
Steps	To test the generateHMAC method with an invalid certificate ID, create an invalid certificate ID. Configure the mock certificateRepository to return Optional.empty() when the findById method is called with the invalid certificate ID. Verify that the generateHMAC method from qrService throws an ApiException, and ensure the response is not null and the certificate ID matches the expected value.
Results	passed
Number	1.3
Purpose	Execute Test for Invalid Signature
Annotation	@Test
Method	testCheckCertivicationWithInvalidSIgnature()
Steps	To test for an invalid signature, create a CheckRequestDTO object with an invalid certificate number and signature. Create a Certificate object with a random ID. Configure the mock certificateRepository to return the Certificate object when the findByNumber method is called. Mock qrService to return a valid signature when the generateHMAC method is called. Verify that the checkCertificate method from certificateService throws an ApiException.
Results	passed

2. Verification Certificate's Unit Testing

   The objective of this testing is to validate the code logic within the certificate verification feature. It entails preparing certificate data for handling by the setup function, which in turn provides essential testing objects like a mocked certificate repository and QR service. The examination focuses on the code logic of the generate HMAC feature, facilitated by the preparation of certificate data managed by the setup function. This data includes requisite objects for testing, such as a mocked certificate repository and QR service. Subsequently, it is utilized across various test scenarios within the testGenerateHMAC function. This function simulates the Hash-Based Message Authentication Code (HMAC) system's response and verifies its conformity with expectations. Following experimentation, unit testing is conducted to validate the successful execution of test scenarios. Presented below is an overview of the created unit testing scenarios

TABLE IV
TEST CASE FOR VERIFICATIONS

Number	2.1
Purpose	Setup Test Environment
Annotation	@BeforEach
Method	setup()
Steps	nitialize mock objects using MockitoAnnotations.initMocks(this)
Results	passed
Number	2.2
Purpose	Execute Test for Invalid Signature
Annotation	@Test
Method	testCheckCertificate_ withSignature_Throws ApiExecption()
Steps	Create a CheckRequestDTO object with an invalid certificate number and signature. Create a Certificate object with a random ID. Configure the mock certificateRepository to return the Certificate object when the findByNumber method is called. Configure the mock qrService to return a valid signature when the generateHMAC method is called. Verify that the checkCertificate method of certificateService throws an ApiException

Results	passed
Number	2.3
Purpose	Data Not Found
Annotation	@Test
Method	testCheckCertificate_With NotExitingCertificate_ ThrowsAPiExecption()
Steps	Create a CheckRequestDTO object with a certificate number that does not exist. Configure the mock certificateRepository to return Optional.empty() when the findByNumber method is called. Verify that the checkCertificate method of certificateService throws an ApiException.
Result	Passed

IV. CONCLUSIONS

The research achieved the successful implementation of the Hash-Based Authentication Code Algorithm within a digital signature generator and validator system using Quick Response Code (QR Code) based web services. This system features PDF generation and certificate delivery via Gmail. Certificate verification is conducted through the certificate number and the decoded result of the QR Code or the certificate signature. The system generates a message digest from certificate data stored in the database using the Hash-Based Authentication Code Algorithm. Database objects are converted into strings and processed to produce the message digest, which is then embedded into the QR Code.

White-box testing through unit testing with JUnit confirmed the success of the system developed with the Springboot framework. The system includes a generator using the Hash-Based Authentication Code Algorithm, a login feature, and a certificate verification feature. All these components passed the tests successfully, demonstrating that the system functions effectively and meets the objectives of the research.

REFERENCES

[1] O. P. Andini, "Cyber Terrorism Criminal Acts in the Perspective of Transnational Organized Crime: International Criminal Law, Global Security Studies," *Unnes Law J. J. Huk. Univ. Negeri Semarang*, vol. 7, no. 2, pp. 333–346, 2021.

[2] Z. Jondong, "Kebijakan Hukum Pidana bagi Tindak Pidana Cyber Terrorism dalam Rangka Pembentukan Hukum Positif di Indonesia," *J. Prefer. Huk.*, vol. 1, no. 2, pp. 21–27, 2020, doi: 10.22225/jph.1.2.2337.21-27.

[3] M. Herlina and R. P. Jati, "The Influence of Cybercrime Against Teenage Angst in Online Media," vol. 343, no. Icas, pp. 379–382, 2019, doi: 10.2991/icas-19.2019.78.

[4] F. Santiago and E. Satoto, "Obstacles and Solutions in Law Enforcement Against the Crime of Electronic Data and Information Falsification," vol. 5, no. 1, pp. 18–26, 2024.

[5] P. Hade and M. Winoto, "Penggunaan Digital Signature Sebagai Keamanan Sistem Informasi," *J. Unikom*, vol. 1, no. 4, 2022, [Online]. Available: https://www.researchgate.net/publication/370098591

[6] B. P. Kavin and S. Ganapathy, "A new digital signature algorithm for ensuring the data integrity in cloud using elliptic curves," *Int. Arab J. Inf. Technol.*, vol. 18, no. 2, pp. 180–190, 2021, doi: 10.34028/IAJIT/18/2/6.

[7] A. Djajadi, K. S. Lestari, L. E. Englista, and A. Destaryana, "Blockchain-Based E-Certificate Verification and Validation Automation Architecture to Avoid Counterfeiting of Digital Assets in Order to Accelerate Digital Transformation," *CCIT J.*, vol. 16, no. 1, pp. 68–85, 2023, doi: 10.33050/ccit.v16i1.2367.

[8] Y. Genc and E. Afacan, "Design and implementation of an efficient elliptic curve digital signature algorithm (ECDSA)," *2021 IEEE Int. IOT, Electron. Mechatronics Conf. IEMTRONICS 2021 - Proc.*, 2021, doi: 10.1109/IEMTRONICS52119.2021.9422589.

[9] B. Angkasa, A. Pambudi, T. Informatika, U. M. Sukabumi, H. Kriptografi, and S. Keamanan, "Implementasi Algoritma Hmac-Sha-256 Implementation of Hmac-Sha-256 Algorithm," vol. 20, no. 2, 2023.

[10] M. A. Berlin, S. Muthusundari, C. S. Anita, D. Rajalakshmi, M. Rajkumar, and R. Dheekshitha, "WITHDRAWN: A HMAC algorithm based secure online transaction system using block chain technology," *Mater. Today Proc.*, no. xxxx, 2020, doi: 10.1016/j.matpr.2020.10.065.

[11] T. D. Prakoso, I. Ernawati, and H. B. Seta, "Penemuan Pola Asosiasi Pada Data Restoran Menggunakan Algoritma Hash Based," *Semin. Nas. Mhs. Ilmu Komput. dan Apl.*, pp. 71–80, 2020.

[12] A. Lorien and T. Wellem, "Implementasi Sistem Otentikasi Dokumen Berbasis Quick Response (QR) Code dan Digital Signature," *J. RESTI (Rekayasa Sist. dan Teknol. Informasi)*, vol. 5, no. 4, pp. 663–671, 2021, doi: 10.29207/resti.v5i4.3316.

[13] J. Yasmin, Y. Tian, and J. Yang, "A First Look at the Deprecation of RESTful APIs: An Empirical Study," *Proc. - 2020 IEEE Int. Conf. Softw. Maint. Evol. ICSME 2020*, pp. 151–161, 2020, doi: 10.1109/ICSME46990.2020.00024.

[14] A. Moore, "auto{API} - A Web-Based Tool for Specification of an API Endpoint to Return JSON Data From an XML Source," *J. Open Res. Softw.*, vol. 9, pp. 1–10, 2021, doi: 10.5334/jors.335.

[15] R. A. Purba, "Application design to help predict market demand using the waterfall method," *Matrix J. Manaj. Teknol. dan Inform.*, vol. 11, no. 3, pp. 140–149, 2021, doi: 10.31940/matrix.v11i3.140-149.

[16] N. Dwivedi, D. Katiyar, and G. Goel, "A Comparative Study of Various Software Development Life Cycle (SDLC) Models," *Int. J. Res. Eng. Sci. Manag.*, vol. 5, no. 3, pp. 141–144, 2022, [Online]. Available: https://www.ijresm.com

[17] J. Fadillah, Y. Syahidin, E. Gunawan, and J. S. Wijaya, "Design of Information System for Outpatient Emergency Room Eligibility Letter to Support BPJS Claim Reporting," *J. Teknol. Inf. dan Pendidik.*, vol. 16, no. 1, pp. 139–155, 2023, doi: 10.24036/jtip.v16i1.717.

[18] W. Darwin, Z. Zulfadli, I. Yuliady, J. Jusmardi, and M. Deswina, "Designing Web-Based Mess and Dormitory Booking Applications," *J. Teknol. Inf. dan Pendidik.*, vol. 17, no. 1, pp. 46–61, 2023, doi: 10.24036/jtip.v17i1.817.

[19] D. Kurniadi, "Designing and Developing of Learning Class Grouping Applications Base on Genetic Algorithms," *J. Teknol. Inf. dan Pendidik.*, vol. 16, no. 1, pp. 109–126, 2023, doi: 10.24036/jtip.v16i1.694.