Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871

Vol. 7, No. 2, April 2026, Page. 1981-1995
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.2.5484

Document Verification And Authentication By Using Password Based Qr Code Signature With Rsa 2048, Aes Encryption, And Sha-256

Mariskha Tri Adithia*[1], Naomi Elianora[2], Maria Veronica[3]

[1,3]Center for Data Science and Artificial Intelligence System
[1,2,3]Department of Informatics, Universitas Katolik Parahyangan, Indonesia

Email: [1]mariskha@unpar.ac.id

Received: Nov 27, 2025; Revised: Dec 5, 2025; Accepted: Apr 21, 2026; Published: Apr 23, 2026

Abstract

This research presents a secure digital-signature framework for document authentication using QR Codes, combining three modern cryptographic primitives: RSA 2048-bit for digital signing, SHA-256 for document-integrity verification, and password-based AES encryption to protect the signer's private key. The system addresses a recurring limitation in previous QR-Code-based signature schemes—the absence of secure private-key storage—by deriving AES keys from user passwords and salts, ensuring that RSA private keys are never stored in plaintext. A web-based implementation was developed to support user registration, signature generation, and document verification, requiring only a PDF file and the associated password from users. Functional testing demonstrates that the system accurately authenticates signer identities, detects any modification to document content, identifies incorrect document numbers, and rejects invalid or non-signature QR Codes. These results confirm that the combination of RSA 2048, SHA-256 hashing, and password-derived AES encryption effectively ensures confidentiality of private keys while preserving document integrity and authenticity. The approach also prevents common forgery scenarios, including document substitution, unauthorized content changes, and QR Code misuse.

Keywords: *AES encryption, Digital signature, Document verification, Password, QR Code, RSA 2048*

1. PENDAHULUAN

Masa pandemi mendorong berbagai perubahan pada hidup manusia. Mulai dari cara berdiskusi yang awalnya dilakukan secara tatap muka langsung lalu berubah menjadi tatap muka daring, metode belanja yang berubah menjadi belanja *online*, sampai perubahan bentuk dokumen, yang awalnya berupa dokumen *hardcopy*, lalu berubah menjadi *softcopy* atau digital. Untuk dokumen resmi, misalnya surat, metode penandatanganannya pun menjadi berubah. Yang awalnya ditandatangani secara langsung, atau dengan istilah tanda tangan basah, menjadi tanda tangan digital. Bentuk tanda tangan digital ini pun ada berbagai macam, misalnya, tanda tangan basah yang didigitalkan dan tanda tangan dengan *Quick Response Code* (*QR Code)*. *QR Code* ditemukan pada tahun 1994 oleh Denso Wave, sebagai pengembangan dari *bar code*, yang dapat memuat lebih banyak informasi [1]. Selain akhirnya marak digunakan tanda tangan digital, *QR Code* juga digunakan untuk menyimpan alamat *website,* logo perusahaan, dan kode pelacak pada jalur produksi. Misalnya pada makalah [2], [3], [4], [5], [6], [7], *QR Code* diaplikasikan untuk menandatangani dokumen, melacak rantai pasok ikan kering pada usaha kecil, pembayaran dengan aplikasi mobile, dan sistem presensi di sekolah. Pada berbagai aplikasi ini, *QR Code* digunakan salah satunya karena kemudahannya.

Tanda tangan untuk dokumen digital fungsinya tidak berbeda dengan tanda tangan pada dokumen *hardcopy*, yaitu untuk mengotentikasi pengirim atau penandatangan surat dan memastikan bahwa surat tidak dipalsukan. Tujuan ini dapat tercapai dengan mudah pada dokumen *hardcopy*, karena tanda tangan

basah diletakkan langsung pada kertas di mana surat dituliskan. Berbeda dengan dokumen digital yang lebih mudah dipalsukan dan diperbanyak tanpa terdeteksi.

Untuk membangun tanda tangan digital agar dapat mengotentikasi pengirimnya dan memastikan keaslian surat, teknik dari bidang kriptografi dapat digunakan. Untuk dapat mengotentikasi identitas pengirim surat, algoritma tanda tangan digital digunakan [8]. Sedangkan untuk memastikan integritas surat, yaitu memastikan bahwa surat belum diubah isinya, fungsi *hash* dapat digunakan [9], [10], [11].

Salah satu algoritma tanda tangan digital yang dapat digunakan adalah algoritma Rivest, Shamir, Adleman (RSA) [8]. Algoritma ini dibangun oleh Ron Rivest, Adi Shamir, dan Leonard Adleman, pada tahun 1977. Algoritma RSA ini adalah algoritma kunci publik yang digunakan untuk melakukan enkripsi dan dekripsi. Untuk mengenkripsi pesan, digunakan kunci privat yang hanya diketahui oleh pembuat pesan saja, sedangkan untuk mendekripsi pesan, digunakan kunci publik, yang sifatnya tidak rahasia. Algoritma ini dapat juga digunakan untuk membangun tanda tangan digital dan memverifikasinya [12]. Agar aman, maka kunci RSA yang digunakan harus minimal sepanjang 2048 bit. Algoritma Advanced Encryption Standard (AES) sendiri adalah algoritma kriptografi kunci simetri, yang digunakan untuk melakukan enkripsi dan dekripsi [8], [13], [14], [15]. Berbeda dengan RSA, dengan algoritma AES, enkripsi dan dekripsi dilakukan dengan menggunakan kunci yang sama, yaitu kunci privat, yang sifatnya rahasia. AES sendiri ditemukan pada tahun 1997, untuk menggantikan algoritma Digital Encryption Standard (DES) yang kuncinya sudah dapat dipecahkan.

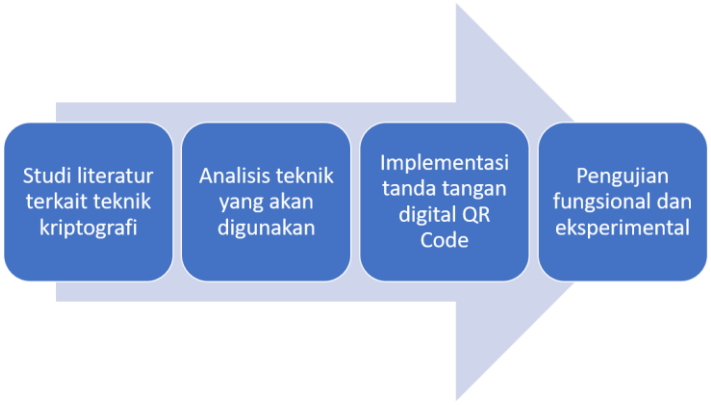
Penandatanganan dokumen dengan menggunakan *QR Code* sudah pernah diteliti sebelumnya. Pada makalah [16], tanda tangan *QR Code* dibangun dengan teknik enkripsi data dokumen dengan algoritma AES. Namun, tidak ada tanda tangan digital dengan teknik kriptografi yang digunakan. Pada makalah [5], tanda tangan *QR Code* digunakan untuk menandatangani sertifikat. Kombinasi teknik yang digunakan adalah tanda tangan digital dengan algoritma RSA, enkripsi dengan algoritma AES, dan fungsi *hash*. Namun, sama dengan makalah [16], algoritma AES adalah bukan algoritma tanda tangan digital. Tanda tangan *QR Code* untuk menandatangani dokumen juga dibahas di makalah [4], [17], namun algoritma yang digunakan bukan algoritma tanda tangan digital; hanya algoritma berbasis *hash* yang bukan fungsi *hash*. Makalah [18] sudah mengimplementasikan algoritma tanda tangan digital dan fungsi *hash* yang baik dengan penjelasan verifikasi tanda tangan yang lengkap. Makalah [19] juga membahas penandatanganan dokumen dengan *QR Code*, di mana algoritma utama yang digunakan adalah RSA, namun fungsi *hash* tidak digunakan untuk memeriksa keaslian isi dokumen; keaslian isi dokumen dilakukan dengan memeriksa secara manual. Kekurangan utama dari berbagai penelitian yang sudah dibahas ini adalah tidak adanya penjelasan maupun mekanisme untuk mengamankan penyimpanan kunci privat. Di beberapa penelitian, kunci privat disimpan dalam bentuk plainteks yang tentutanya tidak aman.

Penelitian ini bertujuan untuk membangun metode untuk membuat tanda tangan digital yang mampu mengotentikasi identitas pengirim surat, memverifikasi keaslian surat, dan mengatasi masalah keamanan penyimpanan kunci privat. Algoritma tanda tangan digital yang digunakan adalah RSA dengan 2048 bit, dan fungsi *hash* yang digunakan adalah Secure Hash Algorithm (SHA) dengan panjang blok 256 bit, yang masih aman dari serangan sampai saat ini [20]. Tentunya kunci privat RSA yang panjang ini akan sulit diingat oleh pengirim surat yang nanti membuat tanda tangan digitalnya sendiri. Kunci privat ini juga tidak mungkin disimpan dalam bentuk plainteks di basis data, karena tidak aman. Salah satu cara mengatasi masalah ini adalah dengan menggunakan kombinasi password dan algoritma AES. Dengan penggunaan password ini, tidak ada kunci rahasia yang disimpan dalam bentuk plainteks. Penggunaan gabungan RSA dan AES untuk pengamanan kunci juga dimuat pada beberapa penelitian sebelumnya [21], [22], [23], [24]. Untuk kemudahan proses verifikasi dan otentikasi bagi pengguna, tanda tangan digital akan berbentuk *QR Code* yang ditempelkan pada dokumen.

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 7, No. 2, April 2026, Page. 1981-1995
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.2.5484

2. METODE

Metode penelitian yang digunakan adalah pertama, dilakukan terlebih dahulu studi literatur terkait tanda tangan digital berbasis password ini. Dari studi ini, juga dilakukan analisis, apakah teknik ini dapat diimplementasikan untuk membangun tanda tangan digital dengan *QR Code*. Selanjutnya, implementasi dilakukan sehingga dihasilkan perangkat lunak berbasis web. Pengguna perangkat lunak ini hanya perlu memasukkan *password* saja, untuk membangun tanda tangan digital *QR Code* untuk suatu dokumen serta memverifikasi dan mengotentikasi dokumen tersebut. Dalam melakukan verifikasi dan otentikasi dokumen, pengguna hanya perlu mengunggah dokumen dengan tanda tangan *QR Code* ke perangkat lunak ini. Perangkat lunak ini selanjutnya diuji, dan jika masih ada kesalahan, dilakukan perbaikan. Metode ini ditunjukkan pada Gambar 1.



Gambar 1. Metode penelitian yang digunakan.

Langkah utama yang dilakukan dalam penelitian ini adalah studi literatur terkait tanda tangan digital berbasis password dan berbagai penelitian sejenis, analisis dan pemodelan masalah, dan pembangunan sistem.

Dari hasil studi literatur, didapat bahwa belum ada sistem sejenis yang dibangun. Gabungan algoritma kriptopgrafi kunci publik dan privat biasanya digunakan, namun tidak ada penjelasan rinci bagaimana kunci privat disimpan dengan aman. Sudah ada solusi untuk memverifikasi dan mengotentikasi dokumen, namun perangkat lunak yang dibangun belum dapat melakukukan verifikasi dan otentikasi tersebut dengan masukan dokumen bertanda tangan digital *QR Code*. Biasanya *QR Code* harus dipindah terpisah. Pada penelitian ini algoritma RSA dipilih sebagai algoritma tanda tangan digital, algoritma AES untuk enkripsi kunci privat RSA di basis data, dan SHA-256 digunakan untuk menghitung nilai *hash* isi surat, yang digunakan untuk memastikan keaslian isi surat.

Algoritma RSA untuk tanda tangan digital terdiri atas tiga tahapan utama, yaitu pembangunan kunci, pembuatan tanda tangan digital, dan verifikasi tanda tangan digital. Untuk pembangunan kunci, mula-mula dipilih dua bilang prima besar, misalnya p dan q. Setelah itu, hitung $n = p \cdot q$ dan $\phi(n) = (p-1)(q-1)$. Selanjutnya, pilih kunci publik e, yaitu sebuah bilangan bulat yang relative prima dengan $\phi(n)$. Terakhir, hitung kunci privat $d = e^{-1} \; mod \; \phi(n)$.

Jika diberikan sebuah pesan m, maka tanda tangan digital s dihitung dengan menggunakan Rumus 1.

$$s = m^d \; mod \; n \tag{1}$$

Untuk memverifikasi tanda tangan, hitung s' dengan menggunakan Rumus 2.

$$s' = s^e \; mod \; n \tag{2}$$

Jika hasil perhitungan s' sama dengan s, maka dapat disimpulkan bahwa tanda tangan valid dan belum ada perubahan dari pesan yang dikirimkan.

Vol. 7, No. 2, April 2026, Page. 1981-1995
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.2.5484

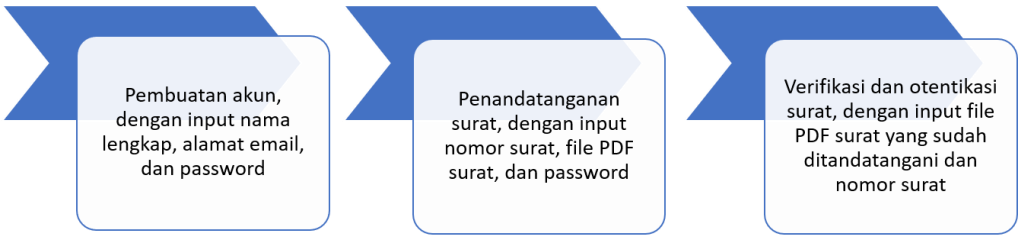
AES adalah algoritma kriptografi kunci privat. Algoritma AES terdiri atas beberapa ronde; banyaknya ronde membutuhkan panjang kunci yang dibutuhkan. Untuk AES dengan 14 ronde, dibutuhkan panjang kunci 256 bit. Tiap ronde AES terdiri atas beberapa operasi yaitu AddRoundKey yaitu operasi XOR antara blok plainteks dengan kunci, SubBytes yaitu substitusi byte dengan menggunakan tabel tertentu, ShitRows yaitu pergeseran melingkar, dan MixColumns yang mengacak data dengan perkalian matriks.

SHA adalah salah algoritma untuk menghitung nilai *hash* kriptografi. Cara kerja SHA adalah dengan memecah input menjadi blok-blok berukuran 512 bit. Tiap blok ini selanjutnya diproses secara berurutan dengan operasi matematis seperti rotasi bit, XOR, fungsi boolean, dan penjumlahan modulo. Operasi ini berlangsung dalam banyak ronde. Nilai yang dihasilkan oleh algoritma ini selalu memiliki panjang tetap. Misalnya untuk SHA-256, panjang outputnya adalah 256 bit.

Pengujian perangkat lunak yang dilakukan adalah pengujian fungsional dan eksperimental. Pada pengujian fungsional, semua fungsionalitas perangkat lunak, seperti membuat akun, *login*, serta pembangunan *QR Code*, diujikan untuk memastikan semua fungsi sudah berjalan dengan baik. Selanjutnya, juga dilakukan pengujian eksperimental yang tujuannya adalah memastikan bahwa *QR Code* pada surat valid dan isi surat tidak diubah. Pengujian dilakukan dengan mengunggah surat berformat PDF yang lengkap dengan nomor surat, untuk diverifikasi perangkat lunak. Respon perangkat lunak terhadap surat yang diunggah selanjutnya dicatat. Berbagai data surat yang digunakan adalah sebagai berikut:

- Surat mengandung *QR Code* yang valid.
- Surat mengandung *QR Code* dengan tanda tangan digital yang benar, namun isi surat sudah dimodifikasi.
- Surat mengandung *QR Code* yang valid, namun nomor surat yang dimasukkan salah.
- Surat mengandung *QR Code* yang valid, nomor surat yang dimasukkan salah, dan isi surat sudah dimodifikasi.
- Surat mengandung *QR Code* yang tidak valid, yang tidak mengandung informasi yang dibutuhkan.
- Surat tidak mengandung *QR Code*.

Dari hasil analisis masalah, dibangun tiga proses utama, yaitu proses pembuatan akun, pembangunan tanda tangan, dan verifikasi serta otentikasi dokumen, yang berjalan secara berurutan (lihat Gambar 2). Ketiga proses utama ini dijelaskan pada sub-sub bagian berikut.



Gambar 2. Alur kerja ketiga proses utama, yaitu pembuatan akun, penandatanganan surat, serta verifikasi dan otentikasi.

2.1. Pembuatan Akun

Pada perangkat lunak yang dibangun, algoritma RSA digunakan untuk pembuatan tanda tangan. Algoritma RSA ini membutuhkan pasangan kunci publik dan privat, di mana kunci publik digunakan untuk membangun tanda tangan, dan kunci privat untuk mengotentikasi tanda tangan. Kedua kunci ini

Vol. 7, No. 2, April 2026, Page. 1981-1995
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.2.5484

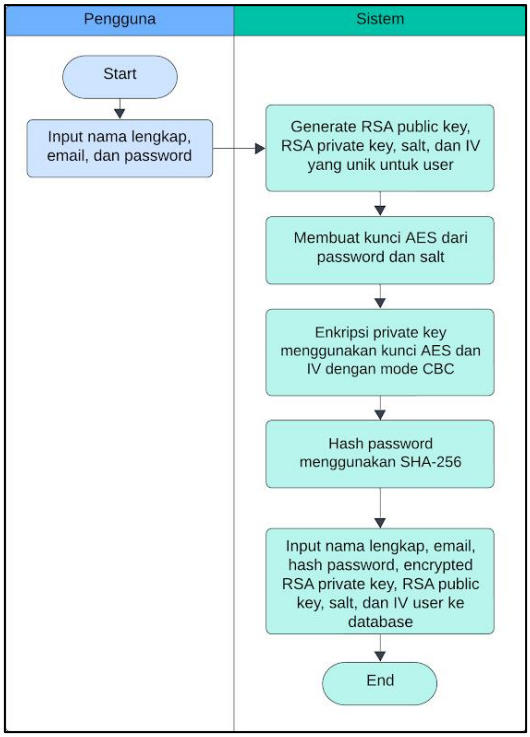
tentu harus spesifik untuk tiap pengguna. Oleh karena itu, pengguna atau penanda tangan surat membutuhkan akun, yang dikaitkan dengan kedua kunci tadi. Informasi yang dibutuhkan untuk membuat akun adalah nama lengkap, alamat *email*, dan *password*.

Seperti yang telah dijelaskan sebelumnya, kunci privat algoritma RSA bersifat rahasia; hanya pemiliknya yang mengetahui. Agar aman, kunci yang digunakan adalah sepanjang 2048 bit. Namun, tentunya sulit bagi pengguna untuk mengingat kunci ini. Maka solusi yang dapat digunakan adalah dengan menyimpan kunci privat ini di basis data. Tentunya kunci ini tidak bisa disimpan sebagai plainteks, namun dienkripsi terlebih dahulu, baru disimpan. Enkripsi kunci privat dapat dilakukan dengan menggunakan algoritma kriptografi kunci simetri, dalam hal ini AES. Agar pengguna tidak perlu menyimpan atau mengingat kunci algoritma AES, kunci ini dibangun dengan menggunakan *pseudorandom number generator*, dengan menggunakan *password* yang sebagai *seed*-nya.

Proses pembuatan akun diawali dengan pengguna memasukkan nama lengkap, alamat *email*, dan *password*. Selanjutnya, sistem membangun pasangan kunci privat dan publik untuk pengguna ini. Sistem juga membangun kunci AES dengan menggunakan *password* dan *salt* yang berupa bilangan acak [25]. Lalu, sistem mengenkripsi kunci privat dengan menggunakan algoritma AES dan kunci AES yang sudah dibangun serta *Initialization Vector* (IV) yang dibangun acak. IV adalah salah satu parameter dalam algoritma AES yang sifatnya tidak rahasia. Password pengguna juga dihitung nilai *hash*-nya dengan menggunakan algoritma SHA-256. Hal ini dilakukan agar password tetap hanya dapat diketahui oleh pemiliknya, namun tetap dapat diverifikasi oleh sistem. Di tahap akhir, nama lengkap, alamat *email*, nilai *hash password*, kunci privat RSA yang sudah dienkripsi, kunci publik RSA, *salt*, dan IV disimpan di basis data. Seluruh langkah ini dapat dilihat pada Gambar 3.

2.2. Pembuatan Tanda Tangan

Pembuatan tanda tangan digunakan saat pengguna ingin menandatangani suatu dokumen atau surat. Dokumen atau surat harus sudah dibuat terlebih dahulu, dan sudah disimpan dalam format PDF. Langkah-langkah pembuatan tanda tangan dapat dilihat pada Gambar 4.



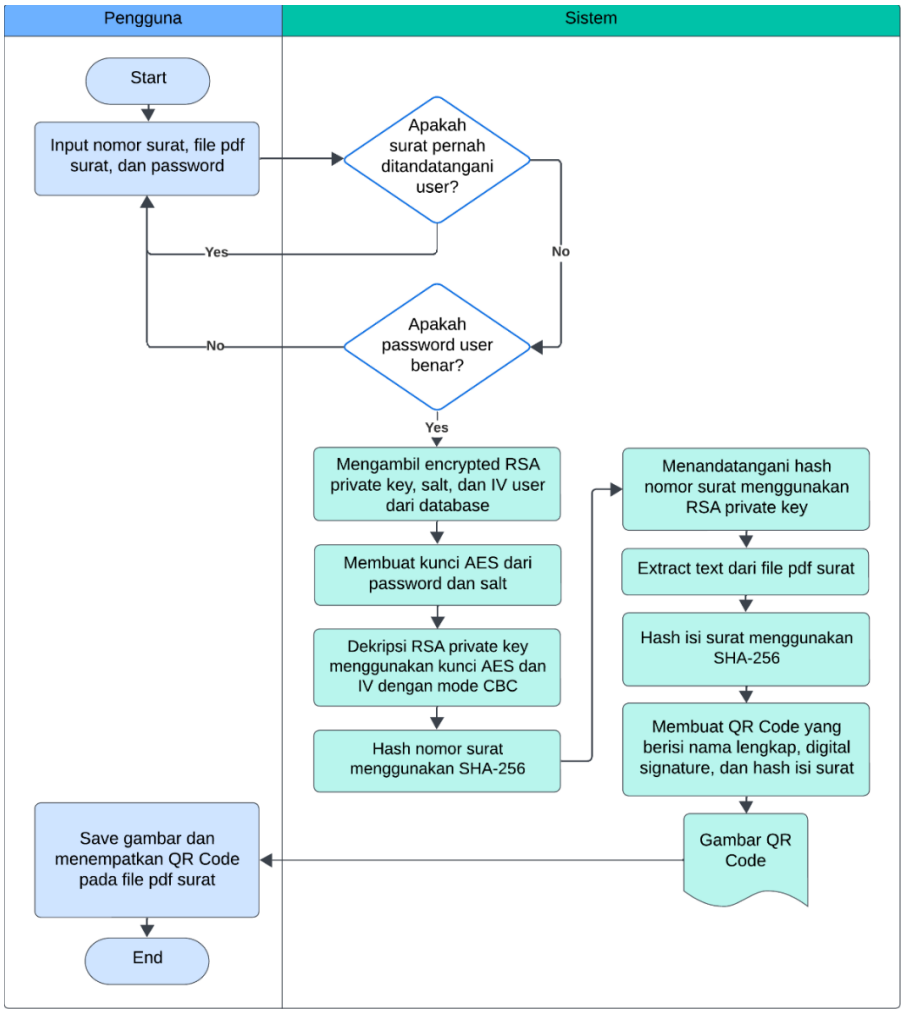
Gambar 3. Proses pembuatan akun pengguna.

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 7, No. 2, April 2026, Page. 1981-1995
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.2.5484

Untuk membuat tanda tangan, terdapat tahap awal untuk mempersiapkan kunci privat RSA. Pada tahap ini, pengguna harus *login* ke sistem terlebih dahulu. Selanjutnya pengguna harus memasukkan nomor surat yang akan ditandatangani, mengunggah surat dalam format PDF, dan memasukkan *password*. Jika *password* benar, sistem mengambil kunci privat RSA yang terenkripsi, IV, dan *salt* dari basis data. Selanjutnya sistem membuat kunci AES dengan menggunakan *password* dan *salt*. Lalu, sistem mendekripsi kunci privat RSA dengan kunci AES yang sudah dibuat dan IV. Saat kunci privat RSA sudah siap, maka tanda tangan siap dibuat.

Tahap selanjutnya adalah mempersiapkan informasi yang akan disimpan pada tanda tangan *QR Code*. Pertama, sistem menghitung nilai *hash* dari nomor surat dengan algoritma SHA-256. Lalu, sistem membuat tanda tangan digital untuk nilai *hash* nomor surat ini dengan menggunakan algoritma RSA dan dengan menggunakan kunci privat RSA yang didapat sebelumnya. Selanjutnya, sistem mengekstraksi isi surat dari file PDF dokumen, dan menghitung nilai *hash* isi surat ini dengan algoritma SHA-256. Langkah ini dilakukan agar keaslian isi surat dapat diverifikasi. Setelah kedua langkah ini dilakukan, tanda tangan *QR Code* siap dibangun.

Pada tahap pembangunan tanda tangan *QR Code* dibangun dengan berisikan nama lengkap penandatangan yang diambil dari basis data, tanda tangan digital nilai *hash* nomor surat, dan nilai *hash* isi surat. Nama lengkap pengguna perlu dimasukkan di dalam *QR Code* agar nanti dapat digunakan untuk memastikan identitas penanda tangan surat. *QR Code* yang dihasilkan ini selanjutnya dapat diletakkan pada surat yang akan ditandatangani, oleh pengguna.



Gambar 4. Proses penandatanganan dokumen atau surat.

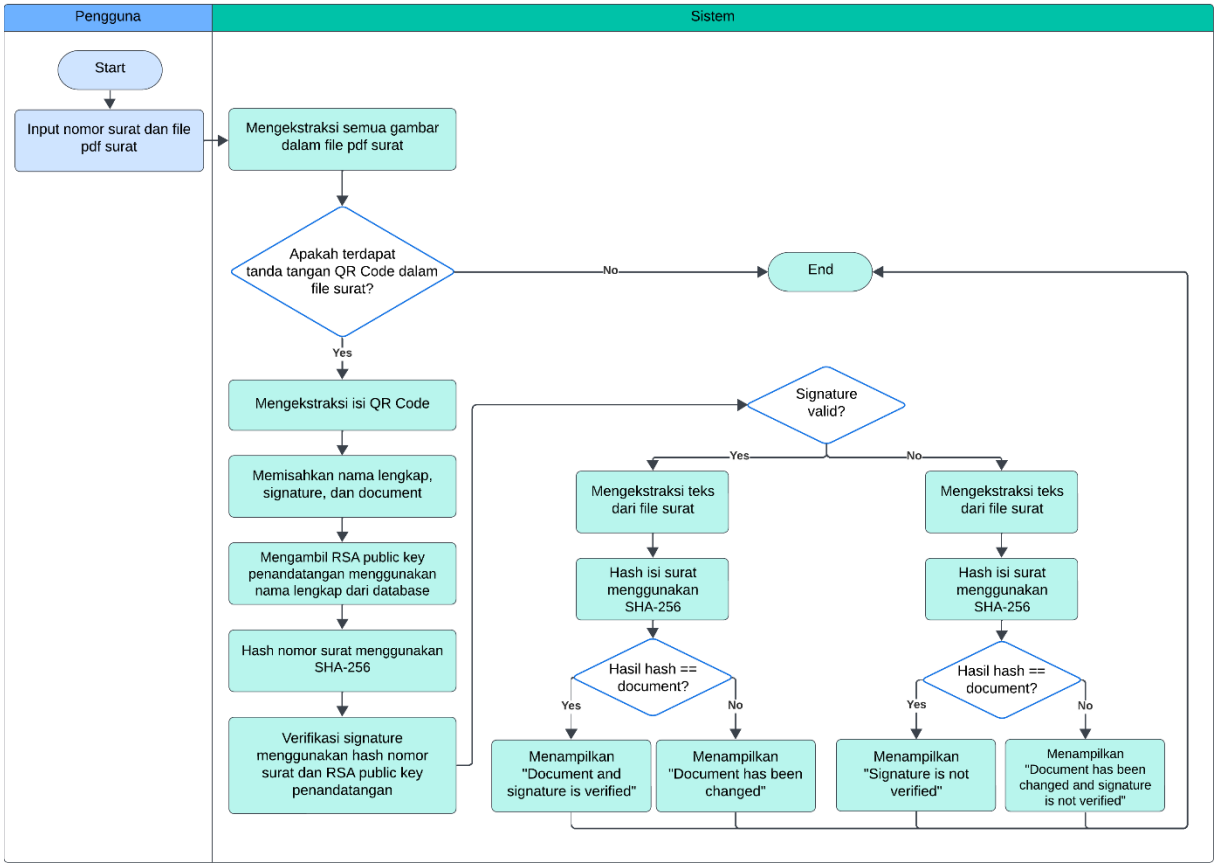
Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 7, No. 2, April 2026, Page. 1981-1995
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.2.5484

2.3. Verifikasi dan Otentikasi Dokumen

Dari proses sebelumnya, telah dihasilkan suatu dokumen yang sudah ditandatangani dengan tanda tangan *QR Code*. Dokumen ini dapat diverifikasi keasliannya dan dipastikan identitas pengirimnya oleh pengguna lain. Langkah-langkah verifikasi dan otentikasi ini diberikan pada Gambar 5.

Tahapan untuk melakukan verifikasi dan otentikasi dokumen terdiri atas tiga tahap, yaitu tahap pencarian gambar *QR Code* pada file PDF surat, penyiapan informasi untuk diverifikasi dan diotentikasi, dan proses verifikasi dan otentikasi itu sendiri.

Untuk memulai tahap pertama, pengguna yang ingin melakukan verifikasi dan otentikasi dokumen tidak perlu memiliki akun. Pengguna hanya perlu membuka website perangkat lunak, lalu memasukkan nomor dan file PDF surat yang sudah bertanda tangan *QR Code*. Sistem lalu akan mengekstraksi seluruh gambar yang ada pada file surat tersebut. Dari seluruh gambar yang didapat, sistem menentukan apakah ada gambar berupa *QR Code*. Jika ada, maka proses diteruskan dengan tahapan penyiapan informasi untuk verifikasi dan otentikasi.



Gambar 5. Proses verifikasi dan otentikasi dokumen bertanda tangan *QR Code.*

Pada tahap kedua, sistem mengekstraksi isi *QR Code* dari dokumen surat, lalu memisahkan nama lengkap, tanda tangan digital, dan nilai *hash* isi surat. Berdasarkan nama lengkap pula, kunci publik RSA diambil dari basis data. Kunci ini nantinya digunakan pada tahap selanjutnya. Lalu sistem juga menghitung nilai *hash* nomor surat yang sebelumnya diinput pengguna. Dari surat yang diunggah sebelumnya, isi surat diekstraksi dan dihitung nilai *hash*-nya. Dari tahap ini, informasi untuk diverifikasi dan diotentikasi sudah siap untuk memasuki tahap terakhir.

Pada tahap ketiga atau yang terakhir ini, dilakukan otentikasi tanda tangan digital dengan menggunakan nilai *hash* nomor surat yang sudah dihitung pada tahap sebelumnya, tanda tangan digital yang dimuat pada *QR Code*, dan kunci publik RSA yang sebelumnya diambil dari basis data. Jika

Vol. 7, No. 2, April 2026, Page. 1981-1995
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.2.5484

otentikasi ini berhasil, maka identitas penandatangan adalah benar. Selanjutnya, hasil perhitungan hasil nilai *hash* isi surat dicocokkan dengan nilai *hash* yang dikandung oleh *QR Code*. Jika sama, maka artinya isi surat juga masih asli.

Dengan melakukan ketiga langkah di atas, keaslian identitas pengirim surat dan isi surat dapat dipastikan. Jika salah satu dari langkah verifikasi atau otentikasi tidak berhasil, maka artinya identitas pengirim surat palsu, atau surat sudah diubah isinya atau dipalsukan.

3. IMPLEMENTASI PERANGKAT LUNAK

Perangkat lunak berbasis web dibangun menggunakan bahasa pemrograman HTML, CSS, dan JavaScript untuk merancang antarmuka pengguna. Untuk bagian server digunakan *framework* Express.js yang merupakan bagian dari *platform* Node.js untuk menangani pengolahan data dan komunikasi antara pengguna dan sistem. Pada bagian server pula, dipakai berbagai library khusus yang berasal dari Node.js [26] dan Node Package Manager (npm) [27], dengan rincian sebagai berikut:

- crypto: Digunakan untuk menghasilkan kunci enkripsi dari password pengguna, menghitung nilai *hash*, membuat *salt*, mengenkripsi, dan mendekripsi kunci privat RSA.
- node-forge: Digunakan untuk menghasilkan pasangan kunci publik dan privat RSA, membuat tanda tangan digital, dan memverifikasi tanda tangan menggunakan kunci publik.
- pdfreader: Digunakan untuk membaca dan mengekstrak teks dari file PDF.
- pdf-lib: Digunakan untuk memuat, memodifikasi dokumen PDF, dan mengakses aliran data mentah dalam objek PDF.
- jimp: Digunakan untuk mengubah format gambar dari PNG ke JPG dan mengatur kualitas gambar *QR Code* sehingga lebih mudah dideteksi.
- qrcode: Digunakan untuk membangun *QR Code,* dan menyimpan informasi tanda tangan digital dalam format gambar.
- qrcode-reader: Digunakan untuk membaca dan mendekode informasi dari *QR Code* dalam bentuk gambar.

Basis data yang digunakan terdiri dari dua tabel, yaitu 'User' untuk menyimpan data pengguna dan 'Digital_Signature' untuk menyimpan informasi terkait tanda tangan digital. Setiap tabel memiliki atribut-atribut yang dijelaskan lebih lanjut pada Tabel 1 dan

Tabel *2*. Pada Tabel 1 dapat dilihat bahwa kunci privat tersimpan di dalam basis data dalam bentuk terenkripsi.

Tabel 1. Tabel ‘User’

Nama Atribut	Tipe	Ukuran	Keterangan
id_user (PK)	int	11	Identifikasi unik pengguna
nama_lengkap	varchar	50	Nama lengkap pengguna
Email	varchar	50	Alamat email pengguna
Password	varchar	500	Password pengguna yang telah di-*hash* dan disimpan dalam format base64
private_key	TEXT	-	Kunci privat pengguna yang disimpan dalam format base64 setelah dienkripsi
public_key	TEXT	-	Kunci publik pengguna yang disimpan dalam format PEM
Salt	varchar	32	Nilai acak yang disimpan dalam format base64
Iv	varchar	32	Nilai acak yang disimpan dalam format base64

Vol. 7, No. 2, April 2026, Page. 1981-1995
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.2.5484

Tabel 2. Tabel 'Digital_Signature'

Nama Atribut	Tipe	Ukuran	Keterangan
id_signature (PK)	int	11	Identifikasi unik tanda tangan digital
no_surat	TEXT	-	Nomor surat yang ditandatangani
Signature	TEXT	-	Tanda tangan digital yang dihasilkan
qr_code	TEXT	-	Gambar *QR Code* yang disimpan dalam format base64
tanggal_ttd	DATE	-	Tanggal tanda tangan digital dihasilkan yang disimpan dalam format YYYY-MM-DD
id_user (FK)	int	11	ID pengguna yang menandatangani surat

Perangkat lunak dijalankan di *localhost* menggunakan XAMPP, yang menyediakan server Apache dan database MySQL. Untuk mengelola basis data, digunakan phpMyAdmin sebagai antarmuka grafis untuk MySQL.

Sesuai dengan yang sudah dijelaskan sebelumnya, pada perangkat lunak ini, kunci privat RSA tersimpan di basis data dalam bentuk terenkripsi dengan menggunakan kunci privat AES. Kunci privat AES ini juga tidak disimpan, tapi dibangkitkan menggunakan password pengguna. Hal ini ditunjukkan pada implementasi kode program pada Gambar 6.

```
//Membuat kunci untuk enkripsi menggunakan password user dan salt
//Menggunakan algoritma PBKDF2 (Password-Based Key Derivation Function 2)
//Salt dan password digunakan untuk menghasilkan kunci derivasi yang unik
//Proses ini dilakukan dengan 100,000 iterasi untuk menambah keamanan ter
//Kunci yang dihasilkan memiliki panjang 32 byte (256 bit)
const key = crypto.pbkdf2Sync(pass_input, salt, 100000, 32, 'sha256');

//menggunakan IV (Initialization Vector) untuk memastikan setiap enkripsi
const iv = crypto.randomBytes(16);

//enkripsi private key mengguankan Algoritma AES-256-CBC (Advanced Encryp
const cipher = crypto.createCipheriv('aes-256-cbc', key, iv);
let encryptedPrivateKey = cipher.update(privateKey, 'utf-8', 'base64');
```

Gambar 6. Potongan kode program untuk membangkitkan kunci provat AES dan mengenkripsi kunci privat RSA.

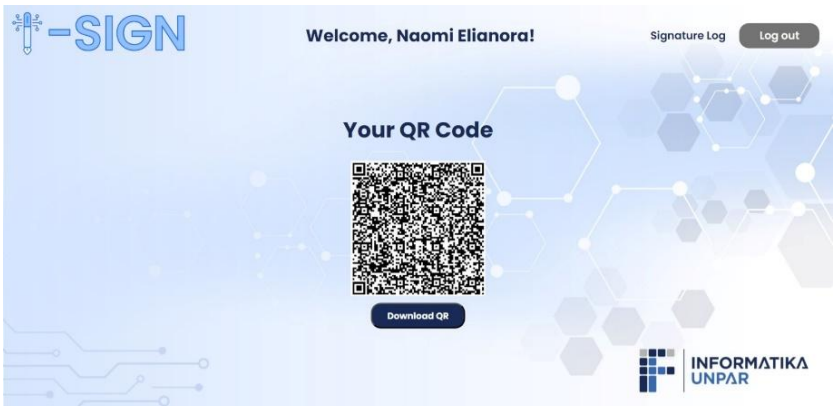
4. HASIL

Perangkat lunak yang dihasilkan memiliki dua antarmuka utama, yaitu halaman untuk membuat tanda tangan dan halaman untuk memverifikasi dan mengotentikasi dokumen. Tangkapan layar keempat antarmuka utama ini diberikan pada Gambar 7.

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 7, No. 2, April 2026, Page. 1981-1995
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.2.5484

Gambar 7. Halaman pembuatan tanda tangan dan verifikasi serta otentikasi dokumen.

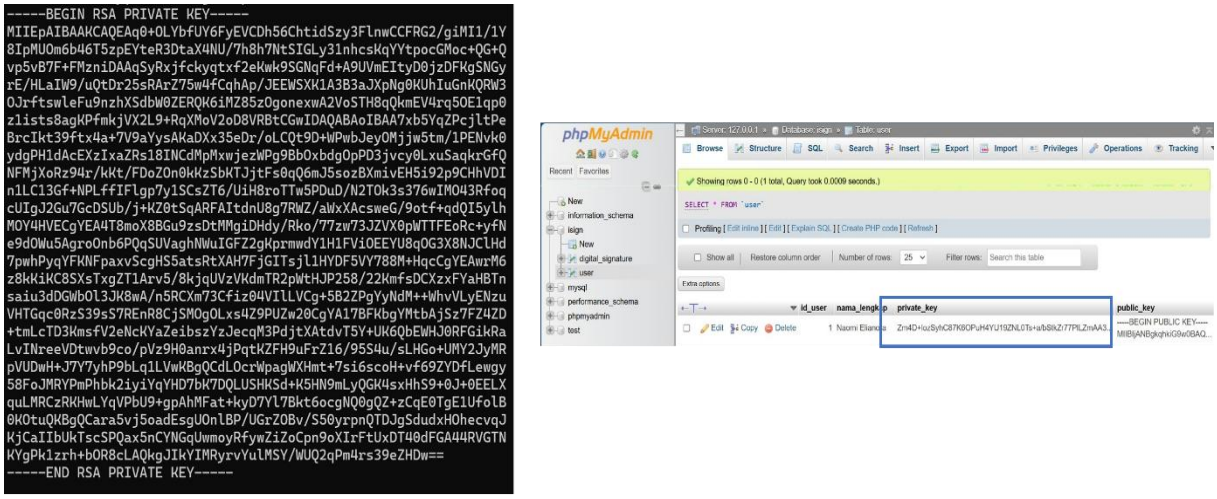
Untuk membuat tanda tangan digital, pengguna diminta memasukkan nomor surat dan password, serta mengunggah file surat dalam format PDF yang akan ditandatangani. Password inilah yang nantinya digunakan sebagai *seed* untuk membangkitkan kunci privat AES, untuk mendekripsi kunci privat RSA di basis data. Nama lengkap dalam hal ini sudah terisi otomatis, diambil dari basis data. *QR Code* yang berhasil dibangun, akan ditunjukkan pada halaman website, dan dapat diunduh (lihat Gambar 8). *QR Code* inilah yang nantinya diletakkan pada surat, sebagai tanda tangan. Verifikasi dan otentikasi dokumen dapat dilakukan untuk pengguna yang sudah mendaftar maupun yang tidak. Seperti ditunjukkan pada Gambar 7**Kesalahan! Sumber referensi tidak ditemukan.**, untuk melakukan verifikasi dan otentikasi dokumen, pengguna diminta untuk memasukkan nomor surat dan surat berformat PDF yang sudah memuat tanda tangan *QR Code*.



Gambar 8. Tanda tangan *QR Code* yang dihasilkan

Sesuai hasil analisis, implementasi basis data juga disesuaikan sehingga tidak ada kunci privat yang disimpan sebagai plainteks. Hal ini ditunjukkan pada Gambar 9 di mana kunci privat diberikan, dan bentuk terenkripsinya yang tersimpan pada basis data.

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 7, No. 2, April 2026, Page. 1981-1995
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.2.5484



(a) (b)

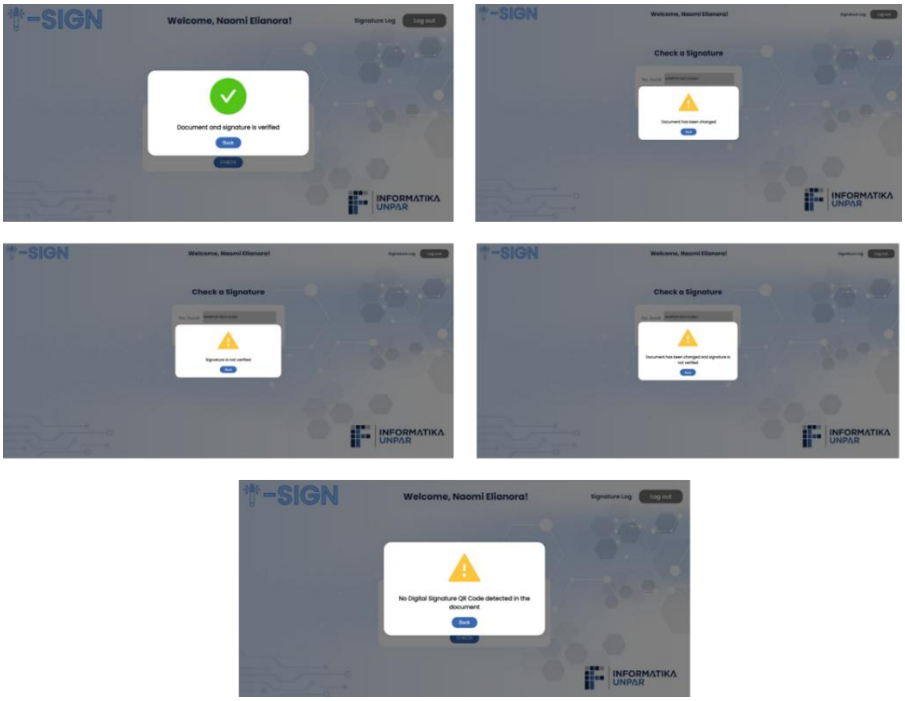
Gambar 9. (a) Kunci privat RSA dan (b) bentuk terenkripsinya di basis data.

Perangkat lunak ini juga sudah diuji keamanannya, yaitu mampu memverifikasi keaslian isi surat dan memastikan identitas penanda tangan surat, yaitu bahwa benar tanda tangan dibuat oleh pihak yang tertera pada surat tersebut. Langkah-langkah pengujian yang digunakan adalah pengguna mengunggah dokumen atau surat, lalu pengguna mengobservasi pesan yang dikeluarkan oleh perangkat lunak. Daftar pengujian yang dilakukan, tujuan, serta hasil yang diharapkan, diberikan pada
Tabel 3.

Tabel 3. Daftar pengujian fungsional perangkat lunak.

No.	Pengujian	Tujuan	Pesan yang Diharapkan
1	Surat mengandung *QR Code* yang valid.	Menunjukkan bahwa tanda tangan digital dan isi surat berhasil diverifikasi.	Document and signature is verified
2	Surat mengandung *QR Code* dengan tanda tangan digital yang benar, namun isi surat sudah dimodifikasi.	Menunjukkan bahwa tanda tangan digital berhasil diverifikasi, namun isi surat dinyatakan sudah berubah.	Document has been changed
3	Surat mengandung *QR Code* yang valid, namun nomor surat yang dimasukkan salah.	Menunjukkan bahwa tanda tangan digital tidak dapat diverifikasi, namun isi surat masih asli.	Signature is not verified
4	Surat mengandung *QR Code* yang valid, nomor surat yang dimasukkan salah, dan isi surat sudah dimodifikasi.	Menunjukkan bahwa tanda tangan digital dan dokumen tidak dapat diverifikasi.	Document has been changed and signature is not verified
5	Surat mengandung *QR Code* yang tidak valid, yang tidak mengandung informasi yang dibutuhkan.	Menunjukkan bahwa verifikasi tidak dapat dilakukan.	No digital signature *QR Code* detected in the document
6	Surat tidak mengandung *QR Code*	Menunjukkan bahwa verifikasi tidak dapat dilakukan.	No digital signature *QR Code* detected in the document

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 7, No. 2, April 2026, Page. 1981-1995
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.2.5484



Gambar 10. Pesan yang dikeluarkan perangkat lunak sebagai hasil pengujian.

Berdasarlan hasil pengujian yang telah dilakukan, semua pesan yang sesuai berhasil dikeluarkan oleh perangkat lunak (lihat Gambar 10). Berdasarkan pengujian pertama pada

Tabel 3, perangkat lunak mampu untuk memverifikasi dan mengotentikasi dokumen. Lalu, berdasarkan hasil pengujian kedua, perangkat lunak sudah mampu mendeteksi jika isi surat yang diunggah tidak sesuai dengan isi surat tang ditandatangani dengan *QR Code*. Artinya, perangkat lunak sudah mampu memastikan integritas surat. Dari hasil pengujian ketiga, perangkat lunak sudah mampu mendeteksi jika nomor surat yang akan diverifikasi tidak sesuai dengan nomor surat yang sudah ditandatangani. Perangkat lunak juga sudah mampu mendeteksi jika *QR Code* yang diletakkan pada file PDF bukanlah tanda tangan *QR Code*, serta jika file PDF surat tidak mengandung *QR Code* sama sekali. Hal ini ditunjukkan dari hasil pengujian kelima dan keenam.

5. DISKUSI

Berdasarkan pengujian eksperimental, kecurangan utama untuk memalsukan tanda tangan atau mengubah isi surat, dapat dihindari. Tiga contoh kasus terkait kecurangan ini dapat dijelaskan sebagai berikut:

- Seorang pengguna meletakkan tanda tangan *QR Code* pada file PDF surat dengan nomor surat yang berbeda dengan nomor surat pada tanda tangan *QR Code*. Pada saat PDF surat bertanda tangan *QR Code* diupload untuk diverifikasi, dan nomor surat berbeda ini dimasukkan ke perangkat lunak, verifikasi dan otentikasi dokumen akan gagal karena tanda tangan digital dan nilai *hash* isi surat tidak cocok.
- Seorang pengguna meletakkan tanda tangan *QR Code* pada file PDF surat yang berbeda dengan PDF surat yang digunakan untuk membangun tanda tangan *QR Code*. Verifikasi dan otentikasi surat juga akan gagal karena nilai *hash* isi surat akan tidak cocok.
- Seorang pengguna meletakkan tanda tangan *QR Code* pada file PDF surat yang sudah ia ganti nama penandatangannya. Verifikasi juga akan gagal karena nilai *hash* isi surat akan tidak cocok

Dari pengujian dan contoh kasus yang diberikan, dapat disimpulkan bahwa perangkat lunak yang dibangun dapat menjaga integritas dokumen yang ditandatangani, dapat mengotentikasi atau memastikan kebenaran identitas penandatangannya, dan dapat mendeteksi kecurangan.

Tabel 4. Analisis komparasi metode usulan dengan metode pada penelitian serupa sebelumnya.

No	Kriteria	Penelitian Sebelumnya					Metode Usulan
		[5]	[16]	[17]	[18]	[19]	
1	Penggunaan teknik tanda tangan digital yang sesuai	Tidak	Tidak	Tidak	Ya	Tidak	Ya
2	Penggunaan teknik pemeriksaan keaslian data yang sesuai	Ya	Tidak	Ya	Ya	Tidak	Ya
3	Keamanan penyimpanan kunci	Tidak	Tidak	Tidak	Tidak	Tidak	Ya
4	Penggunaan varian algoritma yang kuat	Tidak	Tidak	Ya	Ya	Tidak	Ya

Dari hasil implementasi, juga ditunjukkan bahwa tidak ada kunci privat yang disimpan pada basis data; kunci privat tersimpan dalam bentuk terenkripsi dengan algoritma AES. Untuk dapat digunakan, kunci privat, yaitu kunci privat RSA, harus didekripsi terlebih dahulu dengan kunci AES. Kunci AES ini pun tidak pernah disimpan, namun dibangkitkan dengan menggunakan password sebagai *seed*, setiap kali pengguna ingin membuat tanda tangan digital. Dengan cara ini, kunci tersimpan dengan aman dan perangkat lunak tetap dapat digunakan dengan mudah.

Saat dibandingkan dengan penelitian sejenis sebelumnya, kontribusi utama penelitian ini adalah pada keamanan penyimpanan kunci, yang tidak ditangani atau dibahas sama sekali pada penelitian sebelumnya. Selain itu, kontribusi lain adalah pada kesesuaian penggunaan gabungan teknik kriptografi yang digunakan: tanda tangan digital dengan algoritma tanda tangan digital RSA dan pemeriksaan keaslian isi dokumen dengan menggunakan fungsi hash SHA. Varian algoritma yang digunakan juga merupakan varian yang masih paling kuat saat ini, yaitu RSA 2048 bit dan SHA-256. Perbandingan lengkap antara hasil penelitian ini dengan beberapa hasil penelitian sebelumnya, diberikan pada **Kesalahan! Sumber referensi tidak ditemukan.**.

6. KESIMPULAN

Penelitian ini berhasil membangun sebuah metode dan perangkat lunak berbasis web untuk menghasilkan tanda tangan digital berbentuk *QR Code* yang mampu mengotentikasi identitas penanda tangan serta memverifikasi keaslian isi dokumen. Sistem ini memadukan algoritma kriptografi modern, yaitu RSA 2048-bit untuk tanda tangan digital, SHA-256 untuk memastikan integritas dokumen, serta AES berbasis password untuk mengamankan kunci privat pengguna.

Kontribusi utama sistem ini adalah memberikan solusi lengkap terkait penyimpanan kunci privat secara aman. Pada penelitian ini password dimanfaatkan untuk membuat *seed* kunci privat AES, yang digunakan untuk mengenkripsi dan mendekripsi kunci privat RSA. Dengan kata lain, tidak ada kunci privat yang disimpan dalam bentuk plainteks di basis data.

Vol. 7, No. 2, April 2026, Page. 1981-1995
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.2.5484

Selain itu, sistem ini juga mampu membuat tanda tangan digital dalam bentuk *QR Code*, serta melakukan verifikasi dan otentikasi dokumen hanya dengan mengunggah file PDF bertanda tangan. Hasil implementasi menunjukkan bahwa perangkat lunak mampu:

1. Menghasilkan tanda tangan digital dalam bentuk *QR Code* yang memuat identitas penandatangan, tanda tangan digital nomor surat, dan nilai *hash* isi surat.
2. Melakukan verifikasi dan otentikasi dokumen dengan benar, termasuk mendeteksi perubahan pada isi surat, kesalahan nomor surat, atau adanya *QR Code* yang tidak valid.
3. Mencegah berbagai bentuk kecurangan, seperti pemalsuan isi dokumen, penggantian nomor surat, atau penyalahgunaan *QR Code* pada dokumen lain.

Dengan demikian, perangkat lunak ini efektif untuk memastikan integritas dokumen, keaslian identitas penandatangan, serta mendeteksi upaya pemalsuan, sehingga dapat menjadi solusi praktis untuk proses penandatanganan digital yang aman dan mudah digunakan.

Walaupun sudah bekerja dengan baik, ada dua hal yang belum ditangani pada penelitian ini, yaitu meletakkan tanda tangan *QR Code* pada dokumen berformat PDF secara otomatis dan penanganan verifikasi jika dokumen ditandatangani oleh lebih dari satu pihak. Kedua hal ini akan diteliti lebih lanjut pada penelitian selanjutnya.

CONFLICT OF INTEREST

Penulis menyatakan tidak ada *conflict of interest* dalam pelaksanaan penelitian dan penulisan makalah ini.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Pusat Studi Data Science dan Artificial Intelligence System, Program Studi Informatika, dan Fakultas Sains Universitas Katolik Parahyangan atas dukungan dan fasilitas penelitian yang diberikan. Penulis juga berterima kasih kepada Lembaga Penelitian dan Pengabdian kepada Masyarakat Universitas Katolik Parahyangan atas dukungan proses dan pendanaan yang diberikan.

DAFTAR PUSTAKA

[1] M. Hara, "Development and popularization of QR Code – Code development pursuing reading performance and market forming by open strategy", *Synthesiology*, vol. 12, no. 1, pp.19–27, 2019, doi: 10.5571/syntheng.12.1_19.

[2] A. Jose, S. Prasanna Venkatesan, and B. Kumar, "A QR code-based traceability system for dry fish supply chain of micro and small enterprises in India," *Int. J. Indian Culture and Business Management*, vol. 31, no. 2, pp. 145–149, 2024, doi: 10.1504/ijicbm.2024.136801.

[3] M. Tu, L. Wu, H. Wan, Z. Ding, Z. Guo, and J. Chen, "The adoption of QR Code mobile payment technology during Covid-19: A social learning perspective," *Front. Psychol.*, vol. 12, Feb. 2022, doi: 10.3389/fpsyg.2021.798199.

[4] M. Waqas Ayub, I. Winarno, and A. Sudarsono, "QR Code-based smart document implementation using distributed database and digital signature," *Indonesian Journal of Computer Science*, vol. 13, no. 1, pp. 79–92, 2024, doi: 10.33022/ijcs.v13i1.3673.

[5] A. Indra Irawan, I. Hedi Santoso, Istikmal, and M. Rahayu, "Implementation of QR Code attendance security system using RSA and hash algorithms," *Jurnal Nasional Teknik Elektro dan Teknologi Informasi*, vol. 13, no. 1, pp. 53–59, Feb. 2024, doi: 10.22146/jnteti.v13i1.4395.

[6] Y.-W. Chow, W. Susilo, G. Yang, M. H. Au, and C. Wang, "Authentication and transaction verification using QR Codes with a mobile device," in *International Conference on Security, Privacy and Anonymity in Computation, Communication and Storage*, 2016, pp. 437–451.

[7] M. Černý and M. Gogola, "Potential use of RFID and QR Code in the supply chain based on blockchain and smart contract," *Transportation Research Procedia*, vol. 74, pp. 354–362, 2023, doi: 10.1016/j.trpro.2023.11.155.
[8] Behrouz A. Forouzan, *Cryptography and Network Security*, 1st ed. New Delhi: Tata McGraw-Hill Publishing Company Limited, 2007.
[9] Mark Stamp, *Information Security Principles and Practices*, 2nd ed. San Jose, CA, USA: John Wiley & Sons, 2011.
[10] C. Gilbert and M. Abiola Gilbert, "Exploring secure hashing algorithms for data integrity verification," *International Journal of Multidisciplinary Research and Publications (IJMRAP)*, vol. 7, no. 11, pp. 373–390, 2025, doi: 10.2139/ssrn.5251606.
[11] P. Boyanov, "Practical applications of hash functions MD5, SHA-1, and SHA-256 using various software tools to verify the integrity of files," *Journal Scientific and Applied Research*, vol. 27, no. 1, pp. 120–137, 2024, doi: 10.46687/jsar.v27i1.413.
[12] K. Somsuk, "The development of signing and verification methods for high speed digital signatures on electronic official documents by using RSA cryptography," *Cogent Eng*, vol. 11, no. 1, 2024, doi: 10.1080/23311916.2024.2432513.
[13] T. Diah, A. P. Wardhani, and Y. Asriningtias, "Implementation of AES-256 algorithm in the design of company-based digital document security application," *Journal of Information Technology and Computer Science (INTECOMS)*, vol. 6, no. 2, 2023, doi: 10.31539/intecoms.v6i2.8027.
[14] R. Indrayani, P. Ferdiansyah, and M. Koprawi, "Analisis penggunaan kriptografi metode AES 256 bit pada pengamanan file dengan berbagai format," *Digital Transformation Technology*, vol. 4, no. 2, pp. 1245–1251, Feb. 2025, doi: 10.47709/digitech.v4i2.5457.
[15] T. R. Nur Ridawan and R. H. P. Sejati, "Data security implementation with advanced encryption standard 256 in notary mobile applications," *IJARCCE*, vol. 12, no. 11, Nov. 2023, doi: 10.17148/ijarcce.2023.121109.
[16] A. Gani, P. Suratma, and A. Azis, "Tanda tangan digital menggunakan QR Code dengan metode Advanced Encryption Standard," *Techno*, vol. 8, no. 1, pp. 59–68, 2017, doi: 10.30595/techno.v18i1.1482.
[17] F. Nuraeni, Y. Handoko Agustin, D. Kurniadi, and I. Dewi Ariyanti, "Implementasi skema QR-Code dan digital signature menggunakan kombinasi algoritma RSA dan AES untuk pengamanan data sertifikat elektronik," in *Seminar Nasional Teknologi Informasi, Komunikasi dan Industri (SNTIKI) 12*, 2020, pp. 43–52.
[18] A. Lorien, and T. Wellem, "Implementasi sistem otentikasi dokumen berbasis Quick Response (QR) Code dan digital signature," *Jurnal Resti*, vol. 5, no. 4, pp. 663–671, 2021, doi: 10.29207/resti.v5i1.3316.
[19] K. Yasa, P. Sukarata, G. Putra, I. Nugroho, and I. Astawa, "Secure electronic document with QR Code and RSA digital signature algorithm," in *INSTICC*, Jan. 2023, pp. 1370–1375, doi: 10.5220/0010965600003260.
[20] "LNCS 3006 - Security Analysis of SHA-256 and Sisters." [Online]. Available: http://www.ipa.go.jp/security/enc/CRYPTREC/index-e.html. Accessed: Nov. 27, 2025.
[21] J. Bharti and S. Singh, "A hybrid approach using AES-RSA encryption for cloud data security," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 12, no. 21S, pp. 62–69, 2024, doi: 10.19101/ijatee.2016.317005.
[22] Q. Chang, T. Ma, and W. Yang, "Low power IoT device communication through hybrid AES-RSA encryption in MRA mode," *Sci Rep*, vol. 15, no. 1, Dec. 2025, doi: 10.1038/s41598-025-98905-0.
[23] S. Parikh, R. Jhanwar, and A. Singh, "Hybridization of AES and RSA algorithm in file encryption using parallel computing", in *International Conference on Advanced Communication and Intelligent Systems*, 2023, pp. 281–291.
[24] I. Ifrah and Prof. V. Jadeja, "Hybrid AES–RSA encryption and decryption for secure data transmission," *International Journal of Research Publication and Reviews*, vol. 6, no. 10, pp. 1615–1621, Oct. 2025, doi: 10.55248/gengpi.6.1025.3605.

Vol. 7, No. 2, April 2026, Page. 1981-1995
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.2.5484

[25] B. Sugiantoro, "Analysis of password and salt combination scheme to improve hash algorithm security," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 11, 2019, doi: 10.14569/ijacsa.2019.0101158.
[26] "Node.js v25.2.1 Documentation." [Online]. Available: https://nodejs.org/docs/latest/api/. Accessed: Nov. 27, 2025.
[27] "Node Package Manager Documentation." [Online]. Available: https://docs.npmjs.com/. Accessed: Nov. 27, 2025.