BAB II
TINJAUAN PUSTAKA

2.1 Sistem Informasi

Secara garis besar sistem merupakan suatu kumpulan komponen dan elemen yang saling terintegrasi, komponen yang terorganisir dan bekerja sama dalam mewujudkan suatu tujuan tertentu [12]. Menurut [13] informasi adalah data yang telah dikelola dan diproses untuk memberikan arti dan memperbaiki proses pengambilan keputusan, sehingga sistem informasi memiliki pengertian sebagai suatu sistem yang dibuat oleh manusia untuk mencapai suatu tujuan yaitu menyajikan informasi serta berisi sekumpulan prosedur organisasi yang pada saat dilaksanakan akan memberikan informasi bagi pengambil keputusan dan atau untuk mengendalikan organisasi [13].

2.2 Surat Digital

Surat adalah alat komunikasi tertulis yang berasal dari satu pihak dan ditujukan kepada pihak lain untuk menyampaikan warta [14]. Surat resmi adalah surat tertulis yang dibuat oleh suatu organisasi atau instansi, baik organisasi atau instansi swasta maupun pemerintah yang berisi informasi, ketentuan, atau perintah kerja yang dapat dijadikan pedoman [15].

Dengan perkembangan kemajuan teknologi saat ini, muncul salah satu media yang bernama surat digital. Surat digital adalah sebuah surat yang dibuat atau tulis langsung dengan menggunakan media digital seperti komputer dan telepon seluler berdasarkan fitur yang sudah ada dalam media tersebut [15].

2.3 Autentikasi

Autentikasi merupakan sebuah proses identifikasi yang dilakukan oleh pihak yang satu terhadap pihak yang lain ataupun sebaliknya dengan melakukan berbagai proses identifikasi untuk memastikan keaslian dari informasi yang diterima [16]. Autentikasi adalah suatu langkah untuk menentukan atau memastikan bahwa seseorang (atau sesuatu) adalah autentik atau asli. Melakukan autentikasi terhadap

sebuah objek adalah melakukan konfirmasi terhadap kebenarannya, sedangkan pada suatu sistem komputer, autentikasi umumya terjadi pada saat *login* atau permintaan akses.

Beberapa jenis autentikasi yang biasa digunakan untuk mengamankan sistem komputer antara lain *username* dan *password*, CHAP (*challenge handshake authentication protocol*), *digital certificates*, *one time password* (OTP), *multi-factor authentication*, *mutual authentication, biometrics* dan lain-lain [17].

2.4 *One Time Password* (OTP)

Password atau kata sandi dapat digunakan untuk layanan autentikasi, yaitu layanan yang berhubungan dengan identifikasi, baik mengidentifikasi kebenaran pihak-pihak yang berkomunikasi (*user authentication* atau *entity authentication*) maupun mengidentifikasi kebenaran sumber pesan. Dua pihak yang saling berkomunikasi harus dapat mengautentikasi satu sama lain sehingga ia dapat memastikan sumber pesan. Autentikasi sumber pesan secara implisit juga memberikan kepastian integritas data, sebab jika pesan telah dimodifikasi berarti sumber pesan sudah tidak benar [18].

One Time Password (OTP) adalah sebuah *password* yang hanya berlaku untuk sesi *login* tunggal atau transaksi tunggal [18]. Berbeda dengan penggunaan *password* statis, OTP tidak menggunakan *password* yang sama untuk setiap *login* atau transaksi, sehingga jika pihak yang tidak berkepentingan berhasil merekam *password* OTP yang sudah digunakan maka dia tidak akan dapat menyalahgunakan *password* tersebut karena sudah tidak berlaku lagi. Untuk dapat membuat sebuah password OTP, digunakan salah satu metode kriptografi, yaitu fungsi *hash*, dan untuk pemilihan karakternya dipilih secara acak dengan *Pseudo Random Number Generator* [7].

One Time Password memiliki beberapa batasan, yaitu [19]:

a) Mempunyai masa aktif yang tidak lama

   Password OTP memiliki masa aktif yang tidak lama dikarenakan faktor keamanan yang mempengaruhi hal tersebut. Apabila *password* OTP memiliki masa aktif yang lama atau bahkan tidak memiliki masa aktif sama sekali,

maka *password* OTP tersebut rentan terhadap serangan seperti *brute force* yang sering dihadapi oleh *password* statis yang tidak memiliki jangka waktu

b) Bersifat dinamis

Password OTP bersifat dinamis karena setiap *password* yang dibangkitkan tidak akan sama. *Password* yang dibangkitkan akan selalu berubah ubah dan tidak bersifat terikat kepada informasi *login user*.

c) Hanya fungsional pada konteks tertentu

Untuk melakukan sebuah proses autentikasi diperlukan kode yang bersifat terikat dengan proses dari *server*. Sehingga *password* tidak bisa dipakai untuk proses berbeda yang membutuhkan *challenge code* yang berbeda. Konteks ini hanya berlaku bila *password* dihasilkan dari mode CR.

2.5 Fungsi *Hash*

Fungsi *hash* adalah fungsi yang menerima masukan *string* yang panjangnya sembarang dan mengkonversinya menjadi *string* keluaran yang panjangnya tetap (*fixed*) [20]. Fungsi *hash* dapat menerima masukkan *string* apa saja. Jika *string* menyatakan pesan (*message*), maka sembarang pesan M berukuran bebas dikompresi oleh fungsi *hash* H melalui persamaan [7] :

$$h = H(M) \quad \text{(2.1)}$$

dimana

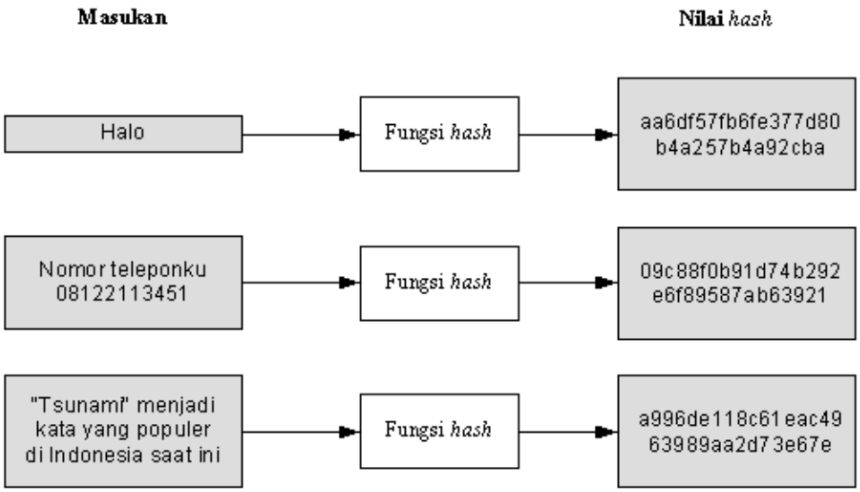
h = *message digest*;

H = fungsi *hash*;

M = pesan berukuran bebas.

Keluaran fungsi *hash* disebut juga nilai *hash* (*hash-value*) atau pesan-ringkas (*message digest*). Dua pesan yang berbeda akan selalu menghasilkan nilai *hash* yang berbeda pula. Sifat-sifat fungsi *hash* [7]:

1) Fungsi H dapat diterapkan pada blok data berukuran berapa saja.
2) H menghasilkan nilai (h) dengan panjang tetap.
3) H(x) mudah dihitung untuk setiap nilai x yang yang diberikan.
4) Untuk setiap h yang diberikan, tidak mungkin menemukan x sedemikian sehingga H(x)=h. Itulah sebabnya fungsi H dikatakan fungsi *hash* satu arah.



Gambar 2.1 Input dan Output Fungsi Hash [19]

Fungsi *hash* biasanya digunakan sebagai metode autentikasi sertifikat web dan metode pencocokan antara *client* dan *server*. Server biasanya akan membandingkan hasil *output* dari fungsi *hash* pengguna dengan *output* dari fungsi *hash* server, hal ini dilakukan agar informasi yang ditukarkan bukan informasi yang sebenarnya dan menghindari dari *intercept* pihak luar [20].

2.6 SHA-512 (*Secure Hash Algorithm 512)*

Algoritma SHA dirancang oleh *National Security Agency* (NSA) dan menjadi standar pemrosesan informasi pada tahun 1993. SHA dibuat berdasarkan fungsi *hash* MD4 dan didesain mirip dengan MD4. Pada tahun 2005, NIST (*National Institute of Standars and Technology*) mengadakan pengumuman agar terjadinya konversi dari penggunaan SHA-1 menjadi SHA-2 pada tahun 2010 [21].

Fungsi hash SHA-512 merupakan fungsi yang menghasilkan *output* sebesar 512 bit dan panjang blok 1024 bit. Terdapat 80 *looping* dalam algoritma ini. Proses *padding message* dilakukan dengan cara yang sama dengan algoritma SHA-1, namun besar hasil akhir pesan menjadi 1024 bit [16].

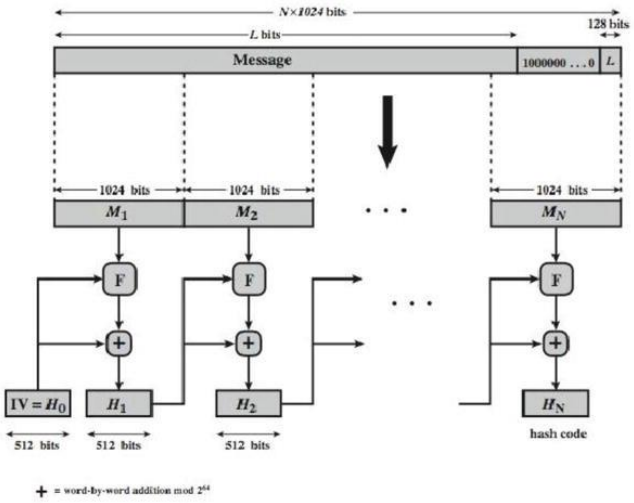
Algoritma SHA-512 menerima masukan string yang kurang dari 2128 bits dan akan menjadikan keluaran berupa 512 bit *message*. Pesan masukkan akan diproses per-1024 bit dan terdiri atas beberapa tahap, yaitu [16]:

a) Menambah jumlah dari bit pesan. Pesan diberi lapisan *padding* agar semua pesan berukuran 1024 bit.
b) Menambah panjang bits. Blok sebanyak 128 bit ditambahkan pada akhir pesan.
c) Inisialisasi *hash buffer*. *Buffer* sebesar 1024 bit digunakan sebagai nilai awal untuk menghasilkan fungsi *hash*. Isi dari 8 nilai awal tersebut adalah sebagai berikut:

Tabel 2.1 Proses Inisialisasi [19]

Huruf	Insialisasi()
A	6a09e667f3bcc908
B	bb67ae8584caa73b
C	3c6ef372fe94f82b
D	a54ff53a5f1d36f1
E	510e527fade682d1
F	9b05688c2b3e6c1f
G	1f83d9abfb41bd6b
H	5be0cd19137e2179

d) Memproses pesan dalam blok 1024 bit

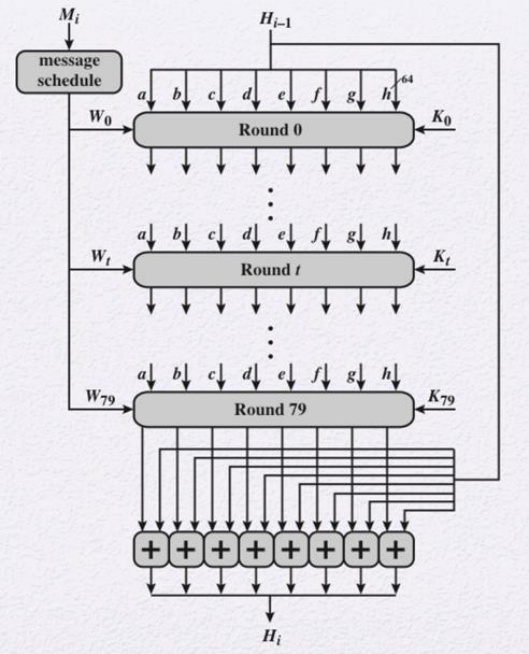


Gambar 2.2 Proses I Algoritma SHA-512 [16]

Babak ini menjadi bagian utama dan akan dilakukan proses pengulangan sebanyak 80 kali. Babak ini direpresentasikan sebagai F pada Gambar 2.2. Setiap babak mendapatkan masukan sebesar 512 bit nilai *buffer*. Variable A,B,C,D,E,F,G,H akan memperbarui nilai *buffer* awal menjadi nilai yang baru. Pada input babak pertama, *buffer* memiliki nilai awal, H*i* − 1. Setiap

babak (*round*) t memakai 64 bit nilai Wt, diperoleh dari 1024 bit blok yang sekarang akan diproses (M*i*). Nilai-nilai ini diperoleh dengan menggunakan message schedule yang akan dijelaskan kemudian. Setiap proses loop juga menggunakan konstanta tambahan Kt, dimana $0 \leq t \leq 79$. Hasil dari proses loop ke 80 menjadi masukan babak pertama (H*i*) untuk dihitung di blok selanjutnya.

e) Setelah semua 1024 bit blok sudah diproses, hasil dari *loop* yang terakhir adalah 512 *bit message*. Secara singkat, proses SHA-512 dapat disimpulkan sebagai berikut.



Gambar 2.3 Proses II Algoritma SHA 512 [16]

f) Setelah semua 1024 bit blok sudah diproses, hasil dari *loop* yang terakhir adalah 512 bit *message*. Secara singkat, proses SHA-512 dapat disimpulkan sebagai berikut [19]:

$$H0 = IV$$

$$Hi = SUM64\ (Hi - 1,\ abcdefghi)$$

$$MD = HN \quad \text{.....} \quad (2.2)$$

dimana :

IV = *Initial Value* (nilai awal) dari *buffer* abcdefgh yang tercantum dalam tahap ketiga

abcdefghi = hasil dari *loop* terakhir dari tahap ke empat

N = jumlah blok dalam pesan

SUM64 = penambahan modulus 264

MD = hasil dari *message*

2.7 *QR Code* (*Quick Response Code*)

QR code adalah gambar berupa matriks dua dimensi yang memiliki kemampuan untuk menyimpan data di dalamnya. *QR code* merupakan evolusi dari kode batang (*barcode*). *Barcode* merupakan sebuah simbol penandaan objek nyata yang terbuat dari pola batang-batang berwarna hitam dan putih agar mudah untuk dikenali oleh komputer [22]. *QR code* merupakan sebuah *barcode* berbentuk dua dimensi, sehingga mereka dapat dibaca dari segala arah di 360. *QR code* dapat menyimpan sampai 4296 karakter alfanumerik [23].

Jadi *QR-code* memiliki kapasitas penyimpanan yang jauh lebih banyak dari *barcode*. Keuntungan lain dari *QR code* adalah dapat dibaca setelah sebagian kerusakan. Berikut merupakan keuntungan dan kerugian menerapkan metode *QR code* menurut [23] yaitu:

a. Keuntungan penggunaan *QR code* :
    1. *QR code* adalah *password* dua dimensi dan dapat dibaca dari segala arah dengan sudut baca 360 derajat.
    2. Kapasitas penyimpanan *QR code* hingga 4296 karakter alfanumerik.
    3. *QR code* dapat dibaca jika terdapat sebagian kerusakan sampai 30%.
    4. Sangat mudah untuk memindai dengan perangkat berbasis kamera *smartphone*.
    5. *QR code* tidak terbaca oleh orang tanpa menggunakan alat pemindai (bisa berupa kamera).
    6. *QR code* dapat menyimpan data dalam satu dimensi kode bar di sepersepuluh ruang.
    7. Hal ini dapat menangani berbagai jenis data seperti angka dan abjad.
b. Kerugian penggunaan *QR code* :

1. Hanya dapat dibaca menggunakan suatu perangkat tertentu (pemindai *QR code*).
2. *QR code* dapat disalin dengan isi kode yang sama.

2.8 Perangkat Lunak Pembangun Aplikasi

2.8.1 Android

Android merupakan sebuah sistem operasi telepon seluler dan komputer tablet layar sentuh (*touchscreen*) yang berbasis Linux [24]. Namun android berubah menjadi *platform* yang begitu cepat dalam melakukan inovasi. *Platform* android terdiri dari sistem operasi berbasis Linux, sebuah GUI (*Graphic User Interface*), *web browser* dan aplikasi *end-user* yang dapat di unduh dan juga para pengembang dapat dengan leluasa berkarya serta menciptakan aplikasi yang terbaik dan terbuka untuk digunakan oleh berbagai macam perangkat [25].

Dalam pemrograman *Java*, ketika menuliskan kode program maka dikompilasi program tersebut dengan menggunakan *Java Compiler* dan di hasilkan *Java Byte Code*. Setelah itu *Java Virtual Mechine* yang akan menjalankan *Java Byte Code* tersebut. Namun, berbeda dengan Android. Di Android, setelah menuliskan kode program maka akan dikompilasi menggunakan *Java Compiler* yang sama, tetapi setelah itu masih perlu dikompilasi ulang dengan menggunakan *Dalvik Compiler* dan *Dalvik Byte Code*. *Dalvik byte code* nantinya akan di eksekusi dalam *Dalvik Virtual Mechine* [25].

2.8.2 Android Studio

Android studio adalah IDE (*Integrated Development Environment*) resmi untuk pengembangan aplikasi android dan bersifat *open source* atau gratis. Peluncuran Android Studio dilakukan oleh *Google* pada 16 Mei 2013, sejak saat itu, Android Studio mengantikan *Eclipse* sebagai IDE resmi untuk mengembangkan aplikasi Android. Android studio sendiri dikembangkan berdasarkan *IntelliJ IDEA* yang mirip dengan *Eclipse* disertai dengan *ADT plugin* (*Android Development Tools*) [18].

2.8.3 *Java*

Java adalah nama sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer yang berdiri sendiri ataupun pada lingkungan jaringan [16]. *Java* merupakan bahasa pemrograman yang berorientasi objek dan dapat dijalankan pada berbagai *platform* sistem operasi. Perkembangan *Java* tidak hanya terfokus pada satu sistem operasi, tetapi dikembangkan untuk berbagai sistem operasi dan bersifat *open source*. Sebagai sebuah bahasa pemrograman, *java* dapat membuat seluruh bentuk aplikasi, *desktop, website* dan lainnya, sebagaimana dibuat dengan menggunakan bahasa pemrograman konvensional yang lain. Untuk membuat aplikasi berbasis *java,* diperlukan *Java Development Kit* (JDK) dan *Java Runtime Environment* (JRE). JDK ini berguna saat anda menulis kode program, sedangkan JRE ini yang memungkinkan sebuah program *java* dapat berjalan di mesin [16].

2.8.4 *Android Software Development Kit* (SDK)

Android Software Development Kit (SDK) adalah tools API (*Application Programming Interface*) yang diperlukan untuk mulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman *Java* [24][5]. Android merupakan bagian perangkat lunak untuk ponsel yang meliputi sistem operasi, *middleware* dan aplikasi kunci yang di *release* oleh *Google*. Saat ini disediakan Android SDK sebagai alat bantu dan API untuk mulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman *Java*. Sebagai *platform* aplikasi yang netral, Android memberi kesempatan untuk membuat aplikasi yang dibutuhkan yang bukan merupakan aplikasi bawaan telepon genggam [5].

2.8.5 XAMPP

XAMPP merupakan paket *PHP* yang berbasis *open source* yang dikembangkan oleh sebuah komunitas *open source* [18][5]. *XAMPP* merupakan suatau program yang didalamnya terdapat beberapa paket program yang sudah dapat langsung dijalankan yaitu *Apache, MYSQL, PHP, File Zila, Phpmyadmin* dan lain-lain.

XAMPP adalah perangkat lunak bebas, yang mendukung banyak sistem operasi, yang merupakan kompilasi dari beberapa program. Fungsi Dari *XAMPP* adalah sebagai *server* yang berdiri sendiri (*local host*), yang terdiri atas program *Apache* HTTP *Server*, *MySQL database*, dan penerjemah bahasa yang ditulis dengan bahasa pemrograman PHP dan *Perl* [21]. Nama *XAMPP* merupakan singkatan dari X (empat sistem operasi apapun), *Apache*, *MySQL*, *PHP* dan *Perl*. Program ini tersedia dalam GNU *General Public License* dan bebas, merupakan *web server* yang mudah digunakan yang dapat melayani tampilan halaman *web* yang dinamis.

2.8.6 *My SQL*

MySQL (*My Structured Query Language*) adalah sebuah program pembuat dan pengelola *database* atau yang sering disebut dengan *DBMS* (*Database Management System*), sifat dari *DBMS* ini adalah *open source* [18]. *MySQL* sebenarnya produk yang berjalan pada *platform* Linux, dengan adanya perkembangan dan banyaknya pengguna, serta lisensi dari *database* ini adalah *Open Source*, maka para pengembang merilis versi *Windows*.

MySQL merupakan program *database* yang mengakses datanya bersifat jaringan, sehingga dapat digunakan untuk aplikasi *Multi User* (banyak pengguna). Kelebihan lain dari *MySQL* adalah menggunakan bahasa *query* (permintaan) standar *SQL* (*Structural Query Languase*). *SQL* adalah suatu bahasa permintaan yang terstruktur, *SQL* telah di standarkan untuk semua program pengakses *database* seperti *oracle*, *PosgresSQL*, *SQL Server* dan lain-lain.

2.8.7 *Hypertext Preprocessor (PHP)*

PHP adalah sebuah bahasa pemrograman *scripting* untuk membuat halaman *web* yang dinamis [16]. *PHP* dikatakan sebagai sebuah *server-side embedded script language* artinya sintak-sintak dan perintah yang kita berikan akan sepenuhnya dijalankan oleh *server* tetapi disertakan pada halaman *HTML* yang seperti biasa. Aplikasi-aplikasi yang dibangun oleh *PHP* pada umumnya akan memberikan hasil pada tampilan *web browser*, tetapi prosesnya secara keseluruhan dijalankan di

server. Terdapat beberapa pandangan dalam mengartikan kata *PHP*, kurang lebih dapat diartiken sebagai *Hypertext Preeprocesor*.

PHP merupakan bahasa pemprograman yang hanya dapat berjalan pada *server* dan hasilnya dapat ditampilkan pada *Client* [19]. *PHP* merupakan produk *Open Source* yang dapat digunakan secara gratis tanpa harus membayar untuk menggunakanya. *PHP* merupakan bahasa standar yang digunakan dalam dunia *Website. PHP* adalah bahasa pemprograman yang berbentuk skrip yang diletakkan didalam *server web*. Jika kita lihat dari sejarah mulainya *PHP* diciptakan dari ide Rasmus Lerdof untuk kebutuhan pribadinya, skrip tersebut sebenarnya dimaksudkan untuk digunakan sebagai keperluan membuat *Website* pribadi, akan tetapi kemudian dikembangkan lagi sehingga menjadi sebuah bahasa yang disebut "*Personal Home Page*".

2.8.8 *Hyper Text Markup Language* (HTML)

Hyper Text Markup Language (HTML) adalah sekumpulan simbol-simbol atau *tag-tag* yang dituliskan dalam sebuah file yang dimaksudkan untuk menampilkan halaman pada *web browser* [31]. *Tag-tag* tersebut memberitahu browser bagaimana menampilkan halaman *web* dengan lengkap kepada pengguna. Tag-tag HTML selalu diawali dengan <x> dan diakhiri dengan </x> dimana x tag *HTML* seperti b, I, u dan sebagainya. Sebuah halaman *website* akan diapit oleh *tag* <html>…….</html> Setiap file HTML selalu berakhiran ekstensi *.htm atau *.html. Jadi jika menemukan *file* dengan ekstensi *.html berarti *file* tersebut adalah berformat HTML.

Sebuah file HTML merupakan file teks biasa yang mengandung tag-tag HTML [31]. HTML merupakan *file* teks, maka HTML bisa dibuat menggunakan teks editor sederhana, misalnya *notepad*. Dapat juga menggunakan HTML *editor* bersifat *visual*, misalnya *dreamweaver*. Untuk mempermudah pembacaan kembali kode-kode HTML, kadang-kadang ditambahkan komentar ke dalam dokumen. Agar komentar tidak bisa dibaca pada browser, maka harus digunakan tanda khusus, yaitu <!—dan diakhiri dengan →.Untuk mencantumkan informasi-informasi itu digunakan tag <ADDRESS>. Umumnya informasi itu diletakkan

pada bagian paling akhir suatu dokumen. HTML menyediakan *tag-tag* untuk membuat sebuah tabel.

2.8.9 *Java Script*

Java Script adalah bahasa pemrograman web yang bersifat *Client Side Programming Language* [17]. *Client Side Programming Language* adalah tipe bahasa pemrograman yang pemrosesannya dilakukan oleh *client*. Aplikasi *client* yang dimaksud merujuk kepada *web browser* seperti Google Chrome, Mozilla Firefox, Opera Mini dan sebagainya. *Java Script* pertama kali dikembangkan pada pertengahan dekade 90'an. Meskipun memiliki nama yang hampir serupa, *Java Script* berbeda dengan bahasa pemrograman *Java*. Menurut [23] dalam [24] untuk penulisannya, *Java Script* dapat disisipkan di dalam dokumen *HTML* ataupun dijadikan dokumen tersendiri yang kemudian diasosiasikan dengan dokumen lain yang dituju. *Java Script* mengimplementasikan fitur yang dirancang untuk mengendalikan bagaimana sebuah halaman web berinteraksi dengan penggunanya.

2.8.10 *Cascading Style Sheets (CSS)*

Cascading Style Sheets (CSS) adalah salah satu bahasa pemrograman desain web (*style sheet language*) yang mengontrol format tampilan sebuah halaman *web* yang ditulis dengan menggunakan bahasa penanda (*markup language*) [25]. Biasanya CSS digunakan untuk mendesain sebuah halaman HTML dan XHTML, tetapi sekarang bahasa pemrograman CSS bisa diaplikasikan untuk segala dokumen XML, termasuk SVG dan XUL [25]. CSS dibuat untuk memisahkan kontek utama (biasanya dibuat dengan menggunakan bahasa HTML dan sejenisnya) dengan tampilan dokumen yang meliputi *layout,* warna dan *font*.

Cara kerja CSS dengan menggunakan dua buah elemen penting untuk pemformatan tampilannya, diantaranya selektor dan deklarator. Dua buah elemen ini berfungsi sebagai penentu format tampilan dan lainnya menempatkan format tampilan tersebut. Deklarator berisi beberapa perintah-perintah CSS untuk menentukan format dari sebuah elemen pada halaman web. Sedangkan selektor

adalah sebuah perintah lanjut dari deklarator dan berfungsi menempatkan format tampilan dari deklarator

2.8.11 *Framework Laravel*

Framework sebagaimana arti dalam Bahasa Indonesia yaitu kerangka kerja dapat diartikan sebagai kumpulan dari *library* (*class*) yang bisa diturunkan, atau bisa langsung dipakai fungsinya oleh modul-modul atau fungsi yang akan dikembangkan [26]. Laravel dirilis dibawah lisensi MIT dengan kode sumber yang sudah disediakan oleh Github. Sama seperti *framework-framework* yang lain, Laravel dibangun dengan konsep MVC (*Model-Controller-View*), kemudian Laravel dilengkapi juga *command line tool* yang bernama "Artisan" yang bisa digunakan untuk *packaging bundle* dan instalasi *bundle* melalui *command prompt* [27].

Beberapa fitur yang dimiliki oleh Laravel adalah sebagai berikut [27]:

1. *Bundles* yaitu sebuah fitur dengan sistem pengemasan modular dan berbagai *bundle* telah tersedia untuk di gunakan dalam aplikasi.
2. *Eloquent ORM* merupakan penerapan PHP lanjutan dari pola "*active record*" menyediakan metode internal untuk mengatasi kendala hubungan antara objek database. Pembangun query Laravel Fluent didukung Eloquent.
3. *Application Logic* merupakan bagian dari aplikasi yang dikembangkan, baik menggunakan Controllers maupun sebagai bagian dari deklarasi Route. Sintaks yang digunakan untuk mendefinisikannya mirip dengan yang digunakan oleh *framework Sinatra*.
4. *Reverse Routing*, mendefinisikan hubungan antara *Link* dan *Route*, sehingga jika suatu saat ada perubahan pada *route* secara otomatis akan tersambung dengan *link* yang relevan. Ketika Link yang dibuat dengan menggunakan nama - nama dari *Route* yang ada, secara otomatis Laravel akan membuat URI yang sesuai.
5. *Restful Controllers*, memberikan sebuah *option* (pilihan) untuk memisahkan logika dalam melayani HTTP GET dan permintaan POST.

6. *Class Auto Loading*, menyediakan otomatis *loading* untuk *class-class* PHP, tanpa membutuhkan pemeriksaan manual terhadap jalur masuknya. Fitur ini mencegah *loading* yang tidak perlu.
7. *View Composers* adalah kode unit *logical* yang dapat dijalankan ketika sebuah *View* di load.
8. *IoC Container* memungkinkan untuk objek baru yang dihasilkan dengan mengikuti prinsip control pembalik, dengan pilhan contoh dan referensi dari objek baru sebagai *Singletons*.
9. *Migrations* menyediakan versi sistem control untuk skema *database*, sehingga memungkinkan untuk menghubungkan perubahan adalah basis kode aplikasi dan keperluan yang dibutuhkan dalam merubah tata letak *database*, mempermudah dalam penempatan dan memperbarui aplikasi.
10. *Unit Testing* mempunyai peran penting dalam framework Laravel, dimana unit testing ini mempunyai banyak tes untuk medeteksi dan mencegah regresi. Unit testing dapat dijalankan melalui itur “artisan command -line”.
11. *Automatic pagination* menyederhanakan tugas dari penerapan halaman, menggantikan penerapan yang manual dengan metode otomatis yang terintegrasi ke Laravel.

2.8.12 *Hypertext Transfer Protocol* (HTTP)

Hypertext Transfer Protocol (HTTP) adalah suatu protokol yang digunakan untuk mengirim dokumen atau halamamn dalam WWW atau *World Wide Web* [36]. Sedangkan pengertian HTTP menurut kamus besar adalah protokol jaringan untuk didistribusikan, kolaboratif, sistem informasi hypermedia. HTTP adalah dasar dari komunikasi data untuk WWW [36]. Dalam pengertian HTTP tersebut, menetapkan bagaimana pesan diformat dan ditransmisikan dan seperti apa respon dari *browser*.

HTTP adalah protokol aplikasi berbasis *client server* sederhana yang dibangun atas TCP (*transmission control protocol*). Sebuah client HTTP biasanya memulai permintaan dengan menciptakan sebuah hubungan ke *port* tertentu di sebuah *server web hosting* tertentu. Umumnya *port* yang digunakan adalah *port* 80. Klien juga sering dikenal dengan *user agent*, sedangkan *server* yang meresponnya

dan juga meyimpan sumber daya seperti berkas HTML dan gambar disebut dengan *origin server*. Selanjutnya sumber yang ingin diakses dengan menggunakan HTTP diidentifikasi dengan menggunakan URL (*Uniform Resource Locator*) dengan skema URL *http* atau *https*.

2.9 Penelitian Sebelumnya (*State of the Art)*

Tabel 2.2 Tabel *State of the Art*

No	Peneliti (Tahun)	Judul	Metode/Alat	Perbedaan dengan Penelitian yang dilakukan Penulis
1.	Nanda Imani Yahya, Safrina Amini (2018)	Pengimplementasian *One Time Password* dan Notifikasi Email Menggunakan Fungsi Hash SHA-512 Berbasis Web Pada Smk *Cyber* Media	*One Time Password* dengan fungsi *Hash* SHA-512	Kode *One Time Password* digunakan untuk proses *log in* pengguna dan menggunakan SMS sebagai media pengiriman OTP (bergantung kepada *server* SMS Gateway) sedangkan pada penelitian ini kode OTP dan fungsi *hash* SHA-512 digunakan untuk verifikasi pengguna yang menindaklanjuti surat dengan media pengiriman menggunakan *email* secara lebih *real time*.
2.	Nani Sarah Hapsari, Yenni Fatman, Isbandi (2020)	Implementasi Metode *One Time Password* pada Sistem Pemesanan Online	Proses enkripsi *Secure Hash Algorithm* 256 (SHA-256)	Metode OTP digunakan untuk aplikasi pemesanan *online* dan menggunakan SHA-256 untuk proses enkripsi sedangkan pada penelitian ini OTP digunakan untuk proses validasi surat menggunakan algoritma SHA-512 yang dimana termasuk fungsi *hash* dengan nilai terpanjang yaitu 512 bit.
3.	Emilio Andi Kriswanto, Fitriyadi (2020)	Implementasi Digital Signature Untuk Validasi Disposisi Surat	Metode *one-way hashing* pada PHP	Mekanisme *Digital Signature* yang digunakan dengan cara menempel tanda tangan asli yang telah dipindai ke aplikasi dan metode *one-way hashing* digunakan untuk proses enkripsi *password* ketika pengguna menindaklanjuti surat sedangkan pada penelitian ini digunakan *QR Code* sebagai bukti digital dan proses validasi surat diamankan menggunakan algoritma SHA-512.
4.	Ibnu Berliyanto G.A, Amir Hamzah, Suwanto Raharjo (2013)	Penerapan *Digital Signature* pada Transkrip Nilai Sebagai Otentikasi Data	Fungsi *hash* algoritma *CRC32*	Metode *hashing* algoritma CRC32 digunakan untuk membangkitkan kode *digital signature* pada transkrip mahasiswa sedangkan pada penelitian ini digunakan algoritma *hash* terbaru yakni SHA-512 untuk membangkitkan kode OTP sebagai bentuk validasi persetujuan surat digital.

5.	Sarfina Adilah, R Rumani M, Marisa W. Paryasto (2017)	Implementasi *Kriptosystem* Menggunakan Metode Algoritma ECC dengan Fungsi Hash SHA-256 Pada Sistem *Ticketing Online*	Algoritma ECC dengan Fungsi Hash SHA-256	Dibangun sistem pemesanan tiket *online* menggunakan algoritma ECC dengan Fungsi Hash SHA-256 untuk validasi berupa tanda tangan digital sedangkan pada penelitian ini sistem menggunakan OTP dengan algoritma SHA-512 untuk proses validasi dan *QR Code* sebagai bukti digital pengganti tanda tangan digital.