

**SKRIPSI**

**PENERAPAN DIGITAL SIGNATURE DENGAN ALGORITMA  
RSA UNTUK MENENTUKAN KEABSAHAN DOKUMEN**

**LAPORAN SKRIPSI**

**(STUDI KASUS FAKULTAS TEKNIK UNIVERSITAS PELITA  
BANGSA)**

Diajukan untuk memenuhi salah satu syarat  
memperoleh Gelar Sarjana Komputer



Disusun oleh:

Sarikhin

311710871

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK**

**UNIVERSITAS PELITA BANGSA**

**BEKASI**

**2021**

**PENERAPAN DIGITAL SIGNATURE DENGAN ALGORITMA  
RSA UNTUK MENENTUKAN KEABSAHAN DOKUMEN  
PENGESAHAN LAPORAN SKRIPSI  
(STUDI KASUS FAKULTAS TEKNIK UNIVERSITAS PELITA  
BANGSA)**

Telah diperiksa dan dishakan  
pada tanggal : .....

## Dosen Pembimbing II

Mengetahui,  
Ketua Program Studi Teknik Informatika

**Aswan Supriyadi Suge, S.E, M.Kom**  
**NIDN. 04260108003**

**LEMBAR PENGESAHAN**

**PENERAPAN DIGITAL SIGNATURE DENGAN ALGORITMA  
RSA UNTUK MENENTUKAN KEABSAHAN DOKUMEN  
PENGESAHAN LAPORAN SKRIPSI**

Disusun oleh:  
Sarikhin  
311710871  
Telah dipertahankan didepan Dewan Penguji  
pada tanggal : .....

Dosen Penguji I

Dosen Penguji II

**Wahyu Hadikristanto, S.Kom., M.Kom  
NIDN.**

**Bambang Hermanto, S.E, M.Kom  
NIDN.**

Dosen Pembimbing I

Dosen Pembimbing II

**Wahyu Hadikristanto, S.Kom., M.Kom  
NIDN.**

**Aswan Supriyadi Suge, S.E, M.Kom  
NIDN. 04260108003**

Mengetahui,  
Ketua Program Studi Teknik Informatika

**Aswan Supriyadi Suge, S.E, M.Kom  
NIDN. 04260108003**

Dekan Fakultas Teknik

**Putri Anggun Sari, S.Pt., M.Si.  
NIDN. 0424088403**

**PERNYATAAN  
KEASLIAN SKRIPSI**

Sebagai mahasiswa Universitas Pelita Bangsa, yang bertanda tangan dibawah ini,  
saya:

Nama : Sarikhin

NIM : 311710871

Menyatakan bahwa karya ilmiah yang berjudul:

“Penerapan Digital Signature Dengan Algoritma RSA Untuk Menentukan  
Keabsahan Dokumen Pengesahan Laporan Skripsi”

merupakan karya asli saya(kecuali cuplikan dan ringkasan yang masing-masing telah saya jelaskan sumbernya dan perangkat pendukung). Apabila dikemudian hari, karya saya disinyalir bukan merupakan karya asli saya, yang disertai dengan bukti-bukti yang cukup, maka saya bersedia untuk membatalkan gelar saya berserta hak dan kewajiban yang melekat pada gelar tersebut. Demikian surat pernyataan ini saya buat dengan sebenarnya.

Dibuat : Bekasi

Pada Tanggal : 16 April 2021

Yang Menyatakan,

Sarikhin

**PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH UNTUK  
KEPENTINGAN AKADEMIS**

Sebagai mahasiswa Universitas Pelita Bangsa, yang bertanda tangan dibawah ini,  
saya :

Nama : Sarikhin

NIM : 311710871

demi mengembangkan Ilmu Pengetahuan, menyetujui untuk memberikan kepada  
Universitas Pelita Bangsa Hak Bebas Royalti Non-Elklusif (*Non  
ExclusiveRoyaltyFreeRight*) atas karya ilmiah yang berjudul :

“Penerapan Digital Signature Dengan Algoritma RSA Untuk Menentukan  
Keabsahan Dokumen Pengesahan Laporan Skripsi”

Beserta perangkat yang diperlukan (bila ada). Dengan Hak Bebas Royalti Non-  
Elklusif ini Universitas Pelita Bangsa berhak untuk menyimpan, mengcopy ulang  
(memperbanyak), menggunakan, mengelolanya dalam bentuk pangkalan data  
(*database*), mendistribusikan dan menampilkan/mempublikasikannya diinternet  
atau media lain untuk kepentingan akademis tanpa perlu meminta ijin dari saya  
selama tetap mencantumkan nama saya sebagai penulis/pencipta.

Saya bersedia untuk menanggung secara pribadi, tanpa melibatkan pihak  
Universitas Pelita Bangsa segala bentuk tuntutan hukum yang timbul atau  
pelanggaran Hak Cipta dalam karya ilmiah saya ini.

Demikian surat pernyataan ini saya buat dengan sebenarnya.

Dibuat : Bekasi

Pada Tanggal : 16 April 2021

Yang Menyatakan,

Sarikhin

# **BAB 1**

## **PENDAHULUAN**

### **1. Latar Belakang Masalah**

Di era digitalisasi seperti sekarang ini mendorong semua aspek berubah menjadi digital, termasuk didalamnya adalah dokumen digital. Dokumen digital menjadi sangat banyak dipakai karena memiliki berbagai macam kelebihan khususnya dalam hal pertukaran dokumen, karena dokumen digital dapat dikirim melalui *email* atau media elektronik lainnya. Salah satu dokumen digital tersebut adalah dokumen pdf. PDF (*portable document file*) di keluarkan oleh perusahaan *Adobe System Inc* pada tahun 1993 .

Universitas Pelita Bangsa merupakan salah satu perguruan tinggi yang dalam kesehariannya menggunakan dokumen pdf, diantaranya adalah dokumen laporan skripsi dengan format pdf. Pada laporan skripsi terdapat halaman dimana halaman tersebut ditujukan untuk membubuhkan tanda tangan. Tidak seperti tanda tangan yang dilakukan pada dokumen biasa, pembubuhan tanda tangan pada dokumen pdf sedikit berbeda. Beberapa orang melakukan penanda tangan pada dokumen digital adalah dengan cara menambah gambar tanda tangan yang kemudian tempatkan pada dokumen pdf tersebut, hal ini merupakan kesalahan karena dengan demikian akan dengan mudah bagi seseorang untuk dengan sengaja memanipulasi tanda tangan tersebut. Seperti halnya tanda tangan pada dokumen fisik, tanda tangan tersebut padat di tiru oleh orang lain dan sulit untuk membuktikan keaslian tanda tangan tersebut.

Masalah keaslian tanda tangan menjadi poin penting, mengingat mudahnya suatu tanda tangan dalam suatu dokumen dapat dimanipulasi dengan sangat mudah dan sulit untuk membedakan antara tanda tangan asli dengan tanda tangan palsu. Pemalsuan tanda tangan ini masih bisa dilakukan karena belum adanya prosedur yang dapat menunjukkan keabsahan tanda tangan pada Universitas Pelita Bangsa.

Berdasar permasalahan keabsahan tanda tangan pada dokumen Laporan Skripsi diatas, mucullah ide untuk membuat sistem sebagai wadah atau saran dalam memberi tanda tangan pada lembar tersebut. Tanda tangan yang dimaksud adalah tanda tangan digital (*digital signature*) yang bertujuan sebagai upaya untuk menjaga keabsahan suatu dokumen. *Digital Signature* adalah salah satu mekanisme untuk menggantikan tanda tangan secara manual pada dokumen kertas. *Digital Signature* memiliki fungsi sebagai penanda pada data yang dapat memastikan bahwa data tersebut adalah data yang sebenarnya. Penanda pada *Digital Signature* ini tidak semata hanya berupa tanda tangan digital, tetapi dapat berupa cap *digital*, *text*, bit, dan gambar. Sistem ini sekaligus dapat melakukan verifikasi terhadap dokumen yang ada apakah dokumen tersebut dalam keadaan asli atau sudah mengalami modifikasi.

## **2. Identifikasi Masalah**

Identifikasi masalah pada penelitian ini adalah :

- a. Penentuan keaslian suatu dokumen masih sulit dilakukan karena belum adanya prosedur untuk membuktikan keaslian suatu tanda tangan pada dokumen Laporan Skripsi.
- b. Belum tersedianya wadah atau prosedur yang digunakan untuk menandatangani dan melakukan validasi keabsahan tanda tangandokumen Laporan Skripsi tersebut.
- c. Suatu tantangan untuk menjaga keaslian suatu tanda tangan karena mudahnya memanipulasi tanda tangan.

## **3. Batasan Masalah**

Berdasarkan latar belakang masalah yang telah diuraikan, penelitian ini berfokus pada proses dan prosedur untuk melakukan penandatanganan pada dokumen Laporan Skripsi dengan menerapkan *Digital Signature* sehingga dapat dilakukan validasi pada tanda tangan tersebut untuk menjaga keabsahannya.

#### **4. Rumusan Masalah**

Rumusan Masalah pada penelitian ini adalah :

- a. Bagaimana menjaga keaslian suatu dokumen dengan menggunakan *Digital Signature*.
- b. Bagaimana tahapan penggunaan *Digital Signature*.
- c. Apakah penggunaan sistem dapat menjadi solusi dengan masalah yang ada.

#### **5. Tujuan penelitian**

Tujuan yang ingin dicapai dari penelitian ini adalah :

- a. Membuat sistem yang dapat menjadi solusi dari permasalahan yang ada.
- b. Menjadikan dokumen dapat di percayai keasliannya

#### **6. Manfaat penelitian**

Manfaat yang diharapkan dari penelitian ini adalah :

- a. Masalah manipulasi menjadi tidak ada karena sudah terdapat sistem yang dapat mendeteksi keaslian suatu tanda tangan.
- b. Proses penandatanganan dokumen menjadi lebih mudah.



## **BAB II**

### **TINJAUAN PUSTAKA DAN LANDASAN TEORI**

#### **2.1. Tinjauan Pustaka**

Berbagai penelitian tentang tanda tangan digital telah dibuat sebelumnya. Jurnal dan penelitian yang membahas kemiripan teori maupun subjek penelitian dijadikan sebagai acuan dalam penelitian ini. Berikut merupakan penelitian terdahulu yang membahas tanda tangan digital :

Pertama, penelitian yang dilakukan oleh Leonardo Refialy, Eko Sedyono dan Adi Setiawan (2015) dalam jurnal Teknik Informatika dan Sistem Informasi Volume 1. Mereka meneliti tentang Pengamanan Sertifikat Tanah Digital Menggunakan Digital Signature SHA-512 dan RSA

Pada penelitian ini fungsi *hash* yang digunakan adalah SHA-512 sebagai algoritma tanda tangan digital karena mempunyai waktu paling tepat dan baik dalam melakukan otentikasi. Hasil dari penelitian ini yaitu dapat mengidentifikasi ada tidaknya perubahan pada dokumen yang telah diberi tanda tangan digital sehingga dapat mengetahui adanya proses manipulasi pada dokumen tersebut, serta tidak terjadinya perubahan yang signifikan terhadap *file* dokumen yang belum diberi tanda tangan digital dan telah diberi sehingga secara kasat mata kedua *file* terlihat sama. Selanjutnya sistem yang dihasilkan dari penelitian ini berupa program *desktop* yang hanya dapat diakses menggunakan computer atau laptop.

Persamaan penelitian sebelumnya dengan penelitian ini adalah sama-sama menggunakan RSA sebagai algoritma enkripsi *public key*. Perbedaan penelitian yang dilakukan terletak pada implementasi sistem dan objek yang digunakan serta algoritma *hash* yang digunakan.

Kedua, penelitian yang dilakukan oleh Egi Cahyo Prabowo dan Irawan Alfrianto (2017) dalam jurnal Ilmiah Komputer dan Informatika (KOMPUTA)

Volume 6. Mereka meneliti tentang Penerapan Digital Signature dan Kriptografi pada Otentikasi Sertifikat Tanah Digital.

Pada penelitian ini fungsi *hash* yang digunakan adalah algoritma SHA-256 dan RSA. SHA-256 dipilih dengan alasan sampai saat ini belum ada yang dapat memecahkan algoritma tersebut dan RSA dipilih karena merupakan system pertama yang sekaligus dapat digunakan untuk *key distribution*, *confidentiality* dan *digital signature*. Hasil dari penelitian ini dapat disimpulkan bahwa penerapan mekanisme pembuatan dokumen digital didalam system dapat memberikan layanan alternative untuk menertibkan dokumen digital dan dapat memberikan layanan keamanan berupa otentikasi dokumen yang diterapkan pada dokumen digital pemohon oleh kepala kanto rsehingga pemohon dapat mengetahui validitas dokumen yang diterima. Persamaan penelitian sebelumnya dengan penelitian ini adalah sama-sama menggunakan algoritma *hash* SHA-256 dan RSA sebagai algortma enkripsi tanda tangan digital. Perbedaan penelitian dilakukan terletak pada implementasi sistem dan objek yang digunakan.

Ketiga, penelitian yang dilakukan oleh Sugiyanto dan Prima Dina Atika (2018) dalam jurnal Cendikia Volume. XVI 2018 yang berjudul Digital Signature dengan Algoritma SHA-1 dan RSA sebagai Autentikasi.

Pada penelitian ini fungsi *hash* yang digunakan adalah SHA-1 karena mempunyai nilai *hash* yang paling kecil. Hasil dari penelitian ini bahwa algoritma *hashing* SHA-1 dan algoritma kriptografi *Rivest Shamir Adleman* (RSA) dapat dikombinasikan dengan baik dalam membuat sebuah *digital signature* pada file pdf dan aplikasi yang dibuat terbukti mampu diandalkan dalam autentikasi file pdf SKPI dari tindak pemalsuan dan modifikasi data serta aplikasi dapat membantu pihak instansi / perusahaan dari rasa khawatirakan modifikasi atau pemalsuan data Surat Keterangan Pendamping Ijasah (SKPI).

Parsamaan sebelumnya dengan penelitian ini dengan adalah sama menggunakan media *website* sebagai implementasi sistem dan objek yang

sama yaitu dokumen pdf. Perbedaan penelitian terletak pada algoritma *hashing* yang digunakan.

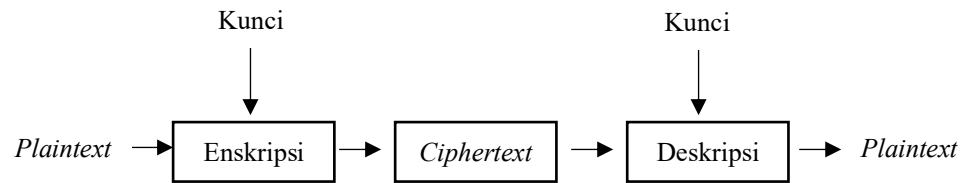
## **2.2. Landasan Teori**

### **2.2.1. Kriptografi**

Kriptografi berasal dari bahasa Yunani, *cryptodan graphia*. *Crypto* artinya *secret* (rahasia) dan *graphia* artinya *writing* (tulisan). Menurut terminologinya, kriptografi adalah ilmu dan seni untuk menjaga keamanan pesan ketika pesan dikirim dari suatu tempat ke tempat lain (Ariyus, 2008). [2]

Kriptografi diperkenalkan oleh orang-orang mesir lewat *hieroglyph* 4000 tahun yang lalu. Dikisahkan pada saat Julius Caesar ingin mengirim pesan rahasia kepada seorang jendral di medan perang melalui seorang kurir. Karena pesan rahasia, Julius Caesar tidak ingin pesan tersebut terbuka di jalan. Untuk mengatasinya ia mengacak pesan hingga menjadi pesan yang tidak dapat dipahami oleh siapapun kecuali jendralnya saja dan sang jendral telah diberitahu sebelum membaca pesan acak tersebut dengan mengganti susunan *alfabet* dari a, b, c yaitu a menjadi d, b menjadi e, dan c menjadi f dan seterusnya. Dari ilustrasi tersebut yang dilakukan Julius Caesar dengan mengacak pesan, disebut dengan enkripsi. Saat sang jendral merapikan pesan yang teracak, disebut dekripsi. Pesan awal yang belum diacak dan pesan yang telah dirapikan disebut *plaintext*, Sedangkan pesan yang telah diacak disebut *chipertext* (Ariyus, 2008). [2]

Selain berdasarkan sejarah yang membagi kriptografi menjadi kriptografi klasik dan kriptografi modern, maka berdasarkan kunci yang digunakan untuk enkripsi dan dekripsi, kriptografi dapat dibedakan lagi menjadi kriptografi Kunci Simetris (*Symmetric-key Cryptography*) dan kriptografi Kunci Asimetris (*Asymmetric-key Criptography*) (Munir, 2006). [2]



Gambar 2.1 Proses Enkripsi/Deskripsi Sederhana [1]

Kriptografi bertujuan untuk memberikan layanan pada aspek-aspek keamanan antara lain:

1. Kerahasiaan (*confidentiality*), yaitu menjaga supaya pesan tidak dapat dibaca oleh pihak-pihak yang tidak berhak,
2. Integritas (*data integrity*), yaitu memberikan jaminan bahwa untuk tiap bagian pesan tidak akan mengalami perubahan dari saat dibuat/dikirim oleh pengirim sampai dengan saat data tersebut dibuka oleh penerima data,
3. Otentikasi (*authentication*), yaitu berhubungan dengan indentifikasi, baik mengidentifikasi kebenaran pihak-pihak yang berkomunikasi maupun mengidentifikasi kebenaran sumber pesan,
4. Nirpenyangkalan (*non repudiation*), yaitu memberikan cara untuk membuktikan bahwa suatu dokumen datang dari seseorang tertentu sehingga apabila ada seseorang yang mencoba mengklaim memiliki dokumen tersebut, dapat dibuktikan kebenarannya dari pengakuan orang tersebut [1].

#### **b. Kriptografi Kunci Simetris (*Symmetric Cryptography*)**

Algoritma ini pengirim dan penerima pesan sudah mengetahui kuncinya sebelum mengirim pesan. Algoritma Simetris (*Symmetric Cryptography*) adalah suatu algoritma dimana kunci enkripsi yang digunakan sama dengan kunci dekripsi, sehingga algoritma ini disebut juga sebagai *single-key algorithm* [2].

Istilah lain untuk kriptografi simetris adalah kriptografi kunci pribadi (*private key cryptography*) atau kriptografi konvensional

(*conventional cryptography*). Keamanan pesan menggunakan algoritma ini tergantung pada kunci. Jika kunci diketahui orang lain, maka orang tersebut dapat melakukan enkripsi dan dekripsi terhadap pesan [1].

Algoritma kriptografi simetris dapat dikelompokkan menjadi 2 (dua) kategori antara lain[1]:

1. *Cipher aliran (stream cipher)*

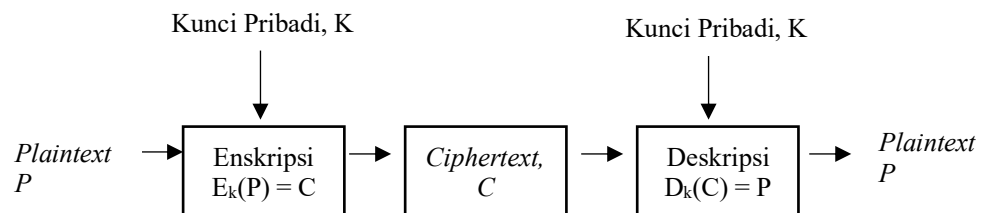
Algoritma kriptografi beroperasi pada *plaintext/ciphertext* dalam bentuk *bit* tunggal yang dalam hal ini rangkaian *bit* dienkripsikan/didekripsikan *bit* per *bit*. *Cipher* aliran mengenkripsi satu *bit* setiap kali.

2. *Cipher blok (block cipher)*

Algoritma kriptografi beroperasi pada *plaintext/ciphertext* dalam bentuk blok *bit*, yang dalam hal ini rangkaian *bit* dibagi menjadi blok-blok *bit* yang penjangnya sudah ditentukan sebelumnya. *Cipher* blok mengenkripsi satu blok *bit* setiap kali.

Algoritma yang memakai kunci simetris diantaranya adalah [2]:

1. Data encryption Standard (DES)
2. RC2, RC4, RC5, RC6
3. International Data Encryption Algorithm (IDEA)
4. Advanced Encryption Standard (AES)
5. One Time Pad (OTP)



Gambar 1.2 Kriptografi Simetris

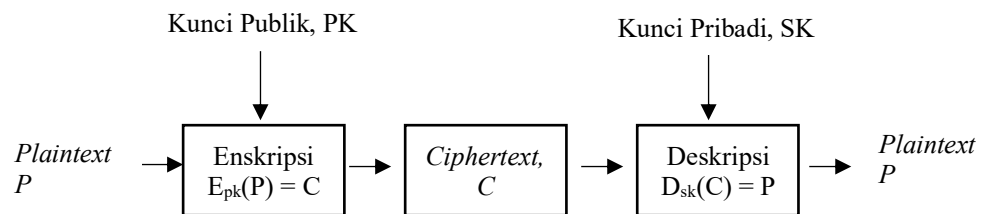
**c. Kriptografi Kunci Asimetris (*Asymmetric Cryptography*)**

Kriptografi asimetris juga disebut dengan kriptografi kunci publik. Kunci yang digunakan enkripsi dan dekripsi berbeda dan

setiap orang yang berkomunikasi mempunyai sepasang kunci yaitu :

1. Kunci umum (*public key*) yaitu kunci yang boleh semua orang tahu.
2. Kunci rahasia (*private key*) yaitu kunci yang dirahasiakan atau diketahui oleh satu orang saja.

Kunci tersebut berhubungan satu dengan yang lain. Walaupun kunci publik sudah diketahui, namun sangat sulit untuk mengetahui kunci *private* yang digunakan. Contoh algoritma kriptografi kunci publik diantaranya, RSA, Elgamal, DSA, dll [2].



Gambar 2.3 Kriptografi Asimetis [1]

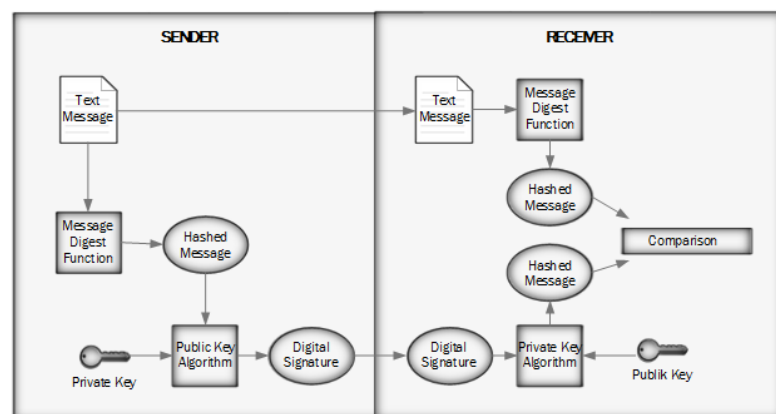
### 2.2.2. Tanda Tangan Digital (*Digital Signature*)

Salah satu konsep pada kriptografi modern adalah *digital signature*. Cara kerja dan kegunaan *digital signature* mirip dengan tanda tangan dalam versi nyata, yaitu untuk memberikan kepastian keaslian dan persetujuan dokumen oleh penanda tangan. Dalam *digital signature*, “tanda tangan” adalah dalam bentuk digital yang digunakan untuk mensahkan sebuah dokumen digital [3].

Prinsip yang digunakan dalam tanda tangan digital ini adalah dokumen yang dikirimkan harus ditandatangani oleh pengirim dan tanda tangan bisa diperiksa oleh penerima untuk memastikan keaslian dokumen yang dikirimkan. Fungsinya adalah untuk melakukan validasi terhadap data yang dikirim. Tanda tangan digital menggunakan algoritma yang disebut dengan istilah *hashing algorithm*. Fungsi tersebut akan menghasilkan sebuah kombinasi karakter yang unik yang disebut *message digest*. dengan cara ini pengirim

bertanggungjawab terhadap isi dokumen dan dapat di cek keaslian dokumen oleh penerima. Keunikannya adalah jika di tengah perjalanan data mengalami modifikasi, penghapusan maupun di sadap diam-diam oleh *hacker* walaupun hanya 1 karakter saja, maka *message digest* yang berada pada si penerima akan berbeda dengan yang dikirimkan pada awalnya. Keunikan lainnya adalah *message digest* tersebut tidak bisa dikembalikan lagi ke dalam bentuk awal seperti sebelum disentuh dengan fungsi algoritma, sehingga disebut sebagai *one-way hash* [3].

Fungsi utama dari tanda tangan digital pada aspek keamanan kriptografi adalah *non-repudiation* atau anti penyangkalan dimana apabila dokumen valid maka pengirim tidak bisa menyangkal bahwa keberadaan dokumen benar dikirim oleh pengirim yang bersangkutan.



Gambar 2.4 Skema Digital Signature [3]

Cara kerja digital signature seperti yang terlihat pada Gambar 4 adalah sebagai berikut [3]:

- Sender melakukan proses hashing algorithm untuk menghasilkan message digest dari sebuah pesan yang terdapat dalam sebuah dokumen yang akan dikirim.
- Setelah dilakukan hashing, Sender melakukan sign terhadap message digest dengan menggunakan kunci publik yang digunakan untuk membentuk digital signature.

- Kemudian Sender mengirimkan digital signature bersama dokumen tersebut kepada Receiver.
- Receiver menerima pesan yang dikirimkan oleh Sender.
- Setelah itu Receiver mengverifikasi pesan yang dikirimkan oleh Sender. Pada proses verifikasi tersebut pesan di hashing terlebih dahulu sehingga menghasilkan message digest dan digital signature akan di unsign menggunakan kunci private. Jika message digest-nya sama, maka pesan ini adalah asli dan pesan berasal dari pengirim yang sebenarnya. Bila pesan telah diubah oleh pihak luar, maka message digest juga ikut berubah.

### 2.2.3. Algoritma Rives Shamir Adleman (RSA)

RSA ditemukan oleh tiga orang yang kemudian disingkat menjadi RSA. Ketiga penemu itu adalah Ron Rivest, Adi Shamir, dan Leonard Adleman. RSA termasuk algoritma asimetris, yang berarti memiliki dua kunci, yaitu kunci publik dan kunci privat. RSA menjadi sistem kriptografi kunci publik yang terpopuler karena merupakan sistem pertama yang sekaligus dapat digunakan untuk *key distribution*, *confidentiality* dan *digital signature* [3].

Pertama kali dipublikasikan pada tahun 1977 oleh ke-3 penemu yang disebut diatas di MIT (Massachusetts Institute of Technology). Karena tergolong algoritma asimetris kunci publik dapat disebarluaskan ke berbagai pihak untuk melakukan enkripsi ataupun dekripsi. Pesan yang sudah terenkripsi dengan kunci publik hanya dapat didekripsi dengan menggunakan kunci privat [2].

Parameter pembangkit kunci RSA dapat dilihat pada tabel 1.

Tabel 2.1 Parameter Pembangkit Kunci RSA

Parameter	Keterangan
$p$	Bilangan prima
$q$	Bilangan prima



$\phi(n)$	Merupakan hasil dari $(p - 1) \times (q - 1)$
$e$	Dengan ketentuan $\gcd(\phi(n), e) = 1$ $\gcd = \text{greatest common divisor}$
$d$	$e^{-1}(\text{mod } \phi(n))$ Menggunakan algoritma <i>extended euclid</i>
<i>Kunci publik</i>	$(e, n)$
<i>Kunci privat</i>	$d$

Langkah-langkah pembangkitan kunci pada RSA [4] adalah:

1. Pilih dua buah bilangan prima sembarang,  $p$  dan  $q$ . untuk memperoleh tingkat keamanan yang tinggi pilih  $p$  dan  $q$  dengan ukuran yang besar, misalnya 1024 *bit*.
2. Hitung  $n = p \cdot q$  (sebaiknya  $p \neq q$ , sebab jika  $p = q$  maka  $n = p^2$  sehingga  $p$  dapat diperoleh dengan menarik akar pangkat dua dari  $n$ ) dimana  $n$  akan digunakan sebagai nilai untuk melakukan modulus pada *public* dan *private key*.
3. Hitung :  $\phi(n) = (p-1)(q-1)$
4. Pilih bilangan integer  $e$  sehingga  $1 < e < \phi(n)$ , dan  $e$  adalah bilangan prima, dimana  $e$  akan digunakan sebagai *private key exponent*.
5. Cari nilai  $d$  sehingga memenuhi:

$$d \equiv e^{-1} \text{ mod } \phi(n), \text{ atau,}$$

$$ed \equiv 1 \text{ mod } \phi(n), \text{ atau,}$$

$$ed \text{ mod } \phi(n) = 1$$

*Private Key* terdiri dari  $n$  sebagai modulus dan  $e$  sebagai eksponen, sedangkan *public key* terdiri dari  $n$  sebagai modulus dan  $d$  sebagai eksponen yang harus dirahasiakan. Nilai eksponen kunci *public* untuk RSA 1024 minimal adalah 65537 untuk menjaga kemanannya. Hubungan antara pesan dapat dituliskan:

$$M^{ed} = M \text{ mod } n$$

Jadi kebutuhan dari algoritma RSA sebelum proses adalah:

- $p, q$ , bilangan prima yang berbeda

- $n = pq$
- $e$ , dimana FPB ( $\phi(n), e$ ) = 1;  $1 < e < \phi(n)$
- $d = e^{-1} \bmod \phi(n)$

Setelah kunci publik  $K_{\text{publik}}$  dibangkitkan oleh pendekripsi, maka sembarang orang dapat menggunakan kunci publik tersebut. Algoritma enkripsi RSA menggunakan fungsi eksponensial dalam modular  $n$ , seperti yang dijelaskan pada Tabel 2.

#### 2.2.4. *Secure Hashing Algorithm (SHA)*

Dalam kriptografi terdapat sebuah fungsi yang digunakan untuk aplikasi keamanan seperti otentikasi dan integritas pesan. Fungsi tersebut adalah fungsi *hash*. Fungsi *hash* merupakan suatu fungsi yang menerima barisan hingga dengan panjang sembarang dan mengkonversinya menjadi barisan hingga keluaran yang panjangnya tertentu. Fungsi *hash* dapat menerima masukan barisan hingga apa saja. Jika barisan hingga menyatakan pesan  $M$ , maka sembarang pesan  $M$  berukuran bebas dimampatkan oleh fungsi *hash*  $H$  melalui persamaan :  $h = H(M)$ .

Keluaran fungsi hash disebut juga nilai *hash* (*hash - value*) atau ringkasan pesan (*message digest* / MD). Pada persamaan di atas,  $h$  adalah nilai *hash* dari fungsi  $H$  untuk masukan  $M$ . Dengan kata lain, fungsi *hash* mengkompresi sembarang pesan yang berukuran berapa saja menjadi nilai *hash* yang panjangnya selalu tetap. Fungsi *hash* merupakan fungsi yang bersifat satu arah (*one-way function*) dimana jika dimasukkan data, maka keluarannya berupa sebuah fingerprint (sidik jari), *Message Authentication Code* (MAC). Fungsi *hash* satu arah adalah fungsi yang bekerja satu arah yaitu pesan yang sudah diubah menjadi ringkasan pesan maka tidak dapat dikembalikan lagi menjadi pesan semula.

Fungsi *hash*  $H$  satu arah mempunyai sifat sebagai berikut :

1. Jika diberikan  $M$  ( $M$  adalah suatu data dalam hal ini berupa pesan), maka  $H(M) = h$  mudah dihitung.
2. Untuk setiap  $h$  yang dihasilkan, tidak mungkin dikembalikan nilai  $M$  sedemikian sehingga  $H(M) = h$ . Fungsi  $H$  disebut fungsi *Hash* satu arah.
3. Jika diberikan  $M$ , tidak mungkin mendapatkan  $M^*$  sedemikian sehingga  $H(M) = H(M^*)$ . Bila diperoleh pesan  $M^*$  semacam ini maka disebut tabrakan (*collision*). Maka sangat sulit menemukan dua pesan yang berbeda yang menghasilkan nilai *hash* yang sama.
4. Tidak mungkin mendapatkan dua pesan  $M$  dan  $M^*$  sedemikian sehingga  $H(M) = H(M^*)$ [5].

SHA didesain oleh *National Security Agency* (NSA) dan dipublikasikan oleh *National Institute of Standards and Technology* (NIST) sebagai *Federal Information Processing Standard* (FIPS) pada tahun 1993 dan disebut sebagai SHA-0, dua tahun kemudian dipublikasikan SHA-1 generasi selanjutnya yang merupakan perbaikan dari algoritma SHA-0. Pada tahun 2002 dipublikasikan empat variasi lainnya yaitu: SHA-224, SHA-256, SHA-384, dan SHA-512, keempatnya disebut sebagai SHA-2 pada sekarang ini.

SHA dinyatakan aman sampai saat ini karena secara komputasi tidak dapat ditemukan isi pesan dari *message digest* yang dihasilkan, dan tidak dapat dihasilkan dua pesan yang berbeda menghasilkan *message digest* yang sama. Setiap perubahan yang terjadi pada pesan akan menghasilkan *message digest* yang berbeda [5].

Algoritma SHA memiliki perbedaan pada ukuran tiap blok, *word* dari data yang digunakan pada saat proses *hashing*, Panjang pesan yang dapat diproses, dan ukuran dari *message digest* yang dihasilkan berbeda-beda sesuai dengan algoritma yang dipakai (ditunjukkan pada Tabel 2.2).

Tabel 2.2 Perbedaan Tiap Variasi Algoritma SHA [5]

Algoritma	Panjang Pesan ( <i>bit</i> )	Ukuran Blok (dalam <i>bit</i> )	Ukuran <i>Word</i> (dalam <i>bit</i> )	Ukuran <i>Message</i> <i>Digest</i> ( <i>bit</i> )	<i>Security</i> ( <i>bit</i> )
SHA-1	$<2^{64}$	512	32	160	80
SHA-256	$<2^{64}$	512	32	256	128
SHA-384	$<2^{128}$	1024	64	384	192
SHA-512	$<2^{128}$	1024	64	512	256

SHA-384 dan SHA-512 memakai 80 konstanta 64 *bit* yang sama, yang ditampung pada variabel  $K_0^{\{512\}}, K_1^{\{512\}}, \dots, K_{79}^{\{512\}}$ . Konstanta dihasilkan dari proses *Fractional parts* dari *cube roots* pada 80 bilangan prima pertama. Dalam *hexadecimal* nilai konstanta tersebut dapat dilihat pada Tabel 2.3 [5].

Tabel 2.3 Konstanta dalam bentuk *hexadecimal*

---

428a2f98d728ae22	7137449123ef65cd	b5c0fbcfec4d3b2f	e9b5dba58189dbbc
3956c25bf348b538	59f111f1b605d019	923f82a4af194f9b	ab1c5ed5da6d8118
d807aa98a3030242	12835b0145706fbc	243185be4ee4b28c	550c7dc3d5ffb4e2
72be5d74f27b896f	80deb1fe3b1696b1	9bdc06a725c71235	c19bf174cf692694
e49b69c19ef14ad2	efbe4786384f25e3	0fc19dc68b8cd5b5	240ca1cc77ac9c65
2de92c6f592b0275	4a7484aa6eae483	5cb0a9dcbd41fbd4	76f988da831153b5
983e5152ee66dfab	a831c66d2db43210	b00327c898fb213f	bf597fc7beef0ee4
c6e00bf33da88fc2	d5a79147930aa725	06ca6351e003826f	142929670a0e6e70
27b70a8546d22ffc	2e1b21385c26c926	4d2c6dfc5ac42aed	53380d139d95b3df
650a73548baf63de	766a0abb3c77b2a8	81c2c92e47edae6	92722c851482353b
a2bfe8a14cf10364	a81a664bbc423001	c24b8b70d0f89791	c76c51a30654be30
d192e819d6ef5218	d69906245565a910	f40e35855771202a	106aa07032bbd1b8
19a4c116b8d2d0c8	1e376c085141ab53	2748774cdf8eeb99	34b0bcb5e19b48a8
391c0cb3c5c95a63	4ed8aa4ae3418acb	5b9cca4f7763e373	682e6ff3d6b2b8a3
748f82ee5defb2fc	78a5636f43172f60	84c87814a1f0ab72	8cc702081a6439ec
90bffffa23631e28	a4506cebd82bde9	bef9a3f7b2c67915	c67178f2e372532b
ca273ecee26619c	d186b8c721c0c207	eada7dd6cde0eb1e	f57d4f7fee6ed178
06f067aa72176fba	0a637dc5a2c898a6	113f9804bef90dae	1b710b35131c471b
28db77f523047d84	32caab7b40c72493	3c9ebe0a15c9bebc	431d67c49c100d4c
4cc5d4becb3e42b6	597f299cfc657e2a	5fcb6fab3ad6faec	6c44198c4a475817

---

Algoritma SHA terdiri dari dua tahap yaitu : *preprocessing* dan proses *hash*. *Prfeprocessing* terdiri dari *padding* pesan, membagi pesan ke dalam *m-bit* blok dan menginisialisasi nilai awal dari *message digest* sebelum dilakukan *hash*. Proses *hash* menghasilkan *message schedule* dari *message* yang telah di-*padding* dan menggunakan jadwal tersebut Bersama fungsi, konstanta dan operasi secara berulang untuk

menghasilkan nilai *hash*. Nilai *hash* yang terakhir dihasilkan dari komputasi akan menjadi *message digest* [6].

#### **2.2.5. Bahasa Pemrograman Python**

Bahasa pemrograman yang interpretative/informative dan multi guna/banyak guna dengan filosofi penyusunan yang terpusat pada tingkat /tahap keterbacaan kode salah satunya adalah Bahasa pemrograman Python [9].

Bahasa pemrograman Python mempunyai fungsi / peran sebagai bahasa yang berfungsi untuk menggabungkan kapabilitas, kemampuan, dengan sintaksis / tata kalimat code yang jelas yang mempunyai fungsionalitas pustaka umum yang besar serta komprehensif/lengkap. Bahasa pemrograman Python juga didukung oleh komunitas yang besar. Bahasa ini banyak digunakan pada programmer karena dikenal dengan bahasa pemrograman yang mudah dipelajari, karena struktur sintak nya yang rapi dan mudah difahami.

Bahasa pemrograman python ini juga men support/mendukung multi paradigm pemrograman, umumnya/pokoknya tidak dibatasi pada pemrograman mengarah kepada objek, pemrograman imperative, dan pemrograman yang fungsional. Beberapa fitur/karakteristik yang terdapat pada Bahasa pemrograman Python ini ialah sebagai bahasa pemrograman dinamis/berfungsi yang dilengkapi dengan manajemen/pengelolaan memori secara otomatis. Bahasa pemrograman python juga umumnya/normalnya dimanfaatkan sebagai suatu bahasa skrip walaupun pada penerapannya penggunaan bahasa pemrograman python ini lebih luas dan membuat konteks pemanfaatan yang umumnya tidak dikerjakan dengan memanfaatkan bahasa skrip. Pemrograman Python ini juga dapat digunakan untuk beragam keperluan/kepentingan dalam

pengembangan suatu perangkat lunak dan dapat berjalan diberbagai platform system operasi [9].

Bahasa Pemrograman Python saat ini dapat dijalankan/digunakan di berbagai platform system operasi, diantaranya adalah sebagai berikut:

- Linux/Unix
- Windows
- Mac OS X
- Java Virtual Machine
- OS/2
- Amiga
- Palm
- Symbian

Bahasa pemrograman python ini dihubungkan dengan berbagai lisensi/sertifikat yang berbeda dari beberapa versi. Kita dapat melihat sejarah di *Python Copyright*. Akan tetapi pada dasarnya Python dapat diperoleh dan dipergunakan secara bebas, terlebih untuk kepentingan komersial [10].

#### **2.2.5.1. Sejarah Python**

Python dikembangkan/ditingkatkan oleh seorang ahli yaitu Guido van Rossum pada tahun 1990 di daerah Sticing Mathematisch Cantrum (CWI) Amsterdam yang merupakan kelanjutan dari bahasa pemrograman ABC. Versi terakhir yang dikeluarkan dari CWI ini adalah versi 1.2.

Pada tahun 1995, Guido pindah ke CNRI di Virginia Amerika sembari terus meneruskan pengembangan Python. Versi terakhir yang dikeluarkan pada tahun itu adalah versi 1.6. tahun 2000, Guido dan para pengembang inti Python lainnya pindah ke BeOpen.com yang merupakan suatu perusahaan komersial dan membangun Be Open Pythin Labs. Python versi

2.0 dikeluarkan oleh Be Open. Setelah mengeluarkan Python 2.0, Guido dan beberapa anggota tim lainnya Python Labs pindah ke Digital Creations [10].

Pada saat ini pengembangan/peningkatan Python terus dilaksanakan oleh sekumpulan pemrogram yang dikoordinir oleh Guido dan Python Software Foundation. Python Software Foundation adalah sebuah organisasi/badan non-profit yang dibentuk menjadi pemegang hak cipta seorang intelektual/ilmuan Python sejak versi 2.1 dengan begitu untuk mencegah supaya Python tidak dimiliki oleh perusahaan komersial [9].

Nama Python diambil oleh ilmuwan Guido sebagai nama bahasa kaeyanya dikarenakan kecintaan Guido pada acara televise yaitu Monty Python's Flying Circus. Oleh karena itu seringkali/kerap kali ungkapan-ungkapan khas dari acara tersebut seringkali tampil di dalam korespondensi/kesesuaian antara pengguna/user Python [10].

#### **2.2.5.2. Aplikasi Penggunaan Python**

Python dapat digunakan di berbagai bidang pengembangan. Berikut ini beberapa aplikasi yang menggunakan python paling populer [9]:

- Website dan Internet

Bahasa pemrograman Python dapat digunakan sebagai server side yang di integrasikan dengan berbagai internet protokol misalnya seperti Email Processing, JSON, FTP, HTML, dan juga IMAP. Selain itu, bahasa pemrograman Python juga memiliki library yang digunakan untuk membangun internet.

- Penelitian ilmiah dan Numerik

Bahasa pemrograman Python dapat digunakan untuk melakukan riset/penelitian ilmiah untuk mempermudah perhitungan/penjumlahan numerik, misalnya seperti penerapan Decision Tree, Navie Bayes, algoritma KNN dan lain-lain.

- Data Science dan Big Data

Bahasa pemrograman Python memungkinkan untuk melakukan proses analisis suatu data dari database big data

#### 2.2.6. Framework Flask

Flask adalah sebuah web framework yang ditulis dengan bahasa python dan tergolong sebagai jenis *microframework*. Flask berfungsi sebagai kerangka kerja aplikasi dan tampilan dari suatu *web*. Dengan menggunakan flask dan bahasa Python, pengembang dapat membuat sebuah *web* yang restruktur dan dapat mengatur *behaviour* suatu *web* dengan lebih mudah [8].

Flask termasuk pada jenis *microframework* karena tidak memerlukan suatu alat atau pustaka tertentu dalam penggunaannya. Sebagian besar fungsi dan komponen umum seperti validasi form, *database*, dan sebagainya tidak terpasang *default* di Flask. Hal ini dikarenakan fungsi komponen-komponen tersebut sudah disediakan oleh pihak ketiga dan Flask dapat menggunakan ekstensi yang membuat fitur seakan diimplementasikan oleh Flask sendiri,

Selain itu, meskipun Flask disebut sebagai *microframework*, bukan berarti Flask mempunyai kekurangan dalam hal fungsionalitas. *Microframework* di sini berarti bahwa Flask bermaksud untuk membuat *core* dari aplikasi ini sesederhana mungkin tetapi tetap dapat dengan mudah ditambahkan. Dengan begitu, fleksibilitas serta kapabilitas dari Flask dapat dikatakan cukup tinggi dibandingkan dengan *framework* lainnya [8].



### 2.2.7. *Library Cryptography Python*

Kriptografi adalah paket yang menyediakan resep kriptografi dan primitif untuk pengembang Python. Ini mendukung Python 3.6+ dan PyPy3 7.2+ [7].

Paket Kriptografi mencakup resep tingkat tinggi dan antarmuka tingkat rendah ke algoritma kriptografi umum seperti cipher simetris, intisari pesan, dan fungsi derivasi kunci. Misalnya, untuk mengenkripsi sesuatu dengan resep enkripsi simetris tingkat tinggi [7].

### 2.2.8. *Library PyHanko Python*

pyHanko merupakan *library* python yang digunakan untuk menambah stempel dan tanda tangan digital dokumen pdf yang berbasis pada Bahasa python. pyHanko Dapat digunakan langsung dengan CLI (*Command line interface*) ataupun dapat dimodifikasi sesuai dengan kebutuhan. puHanko juga dapat digunakan untuk melakukan verifikasi tanda tangan digital pada dokumen pdf [11].

Fitur pertama yaitu menambah stempel pada dokumen pdf, terdapat 2 (dua) jenis stempel yang dapat dibuat pada pyHanko, yaitu [11]:

- a. Stempel Tulisan, hanya terdapat tulisan pada stempel yang di buat dan disertai dengan tanggal stempel
- b. Stempel dengan QR (*Quick Response*), pada stempel terdapat QR yang dapat dipindai.

Fitur kedua adalah menambah tanda tangan digital pada dokumen pdf. Terdapat 2 jenis tanda tangan yang dapat dibuat dengan pyHanko yaitu [11]:

- a. Tanda tangan tidak terlihat namun dapat dibuktikan keberadaannya
- b. Tanda tangan terlihat pada dokumen pdf

### 2.2.9. *Database MYSQL*

Database adalah suatu kumpulan data-data yang disusun sedemikian rupa sehingga membentuk informasi yang sangat berguna. Database terbentuk dari sekelompok data-data yang memiliki jenis/sifat yang sama. Ambil contoh, data-data berupa nama-nama, kelas-kelas, alamat-alamat. Semua data tersebut dikumpulkan menjadi satu menjadi kelompok data baru, sebut saja sebagian data-data mahasiswa. Demikian juga kumpulan dari data-data mahasiswa, data-data dosen, data-data keuangan dan lainnya dapat dikumpulkan lagi menjadi kelompok besar, misalkan data-data fakultas teknik informatika. Bahkan dalam perkembangannya, data-data tersebut dapat berbentuk berbagai macam data, misalkan dapat berupa program, lembaran-lembaran untuk entry (memasukkan) data, laporan-laporan. Kesemuanya itu dapat dikumpulkan menjadi satu yang disebut database [18].

Salah satu bahasa database yang populer adalah SQL. MySQL biasa dibaca mai-es-ki-el atau mai-se-kuel adalah suatu perangkat lunak database relasi (*Relational Database Management System RDBMS*) seperti halnya Oracle, Postgresql, MS SQL dan sebagainya. SQL atau singkatan dari *Structured Query Language* ialah suatu sintaks perintah-perintah tertentu atau Bahasa pemrograman yang digunakan untuk mengelola suatu database. Jadi, MySQL dan SQL tidaklah sama. Singatnya, MySQL ialah perangkat lunak dan SQL adalah bahasa perintah. Ketika dibandingkan antara MySQL dan database lain, maka perlu difikirkan apa yang paling penting sesuai kebutuhan. Apakah tampilan, support, fitur-fitur SQL, kondisi keamanan dalam lisensi, atau masalah harga. Dengan pertimbangan tersebut, MySQL memiliki banyak hal yang bisa ditawarkan, antara lain:

- c. berdasarkan kepentingan, banyak ahli memberikan pendapat bahwa MySQL merupakan server cepat.
- d. MySQL memiliki performa tinggi namun merupakan database yang simple sehingga mudah di-setup dan dikonfigurasi.

- e. MySQL cenderung gratis untuk penggunaan tertentu.
- f. MySQL mengerti Bahasa SQL (*Structured Query Language*) yang merupakan pilihan sistem database modern.
- g. Banyak klien dapat mengakses server dalam satu waktu. Mereka dapat menggunakan banyak database secara simultan.
- h. Database MySQL dapat diakses dari semua tempat di internet dengan hak akses tertentu.
- i. MySQL dapat berjalan dalam banyak varian Unix dengan baik, sebaik seperti saat berjalan di sistem non-Unix.
- j. MySQL mudah didapatkan dan memiliki source code yang boleh disebarluaskan sehingga dapat dikembangkan lebih lanjut.
- k. Dapat dikoneksikan pada Bahasa C, C++, Java, Perl, PHP dan Python.

Jika hal-hal diatas ialah kelebihan yang dimiliki oleh MySQL maka MySQL juga memiliki kekurangan seperti:

- a. Untuk koneksi ke Bahasa pemrograman visual seperti visual basic, delphi, dan foxpro, MySQL kurang mendukung, karena koneksi ini menyebabkan field yang dibaca harus sesuai dengan koneksi dari program visual tersebut. Dan ini yang menyebabkan MySQL jarang dipakai dalam program visual.
- b. Data yang ditangani belum begitu besar.

#### **2.2.10. HTML**

Sebuah bahasa markah untuk membuat halaman web dan bahasa yang digunakannya masih sangat standar seperti salah satu fungsinya untuk membuat tabel, menambahkan objek suara, video dan animasi adalah pengertian dari HTML [12].

Menurut Sibero (2013:19), "*Hypertext Markup Language* atau HTML adalah bahasa yang digunakan pada dokumen sebagai bahasa untuk pertukaran dokumen web [13].

Pengertian di atas dapat disimpulkan bahwa HTML adalah sebuah dokumen yang berisikan tag, beberapa elemen dan atribut untuk menampilkan halaman pada *web browser*.

#### 2.2.11. Pengertian Bootstrap

*Bootstrap* sebuah alat bantu untuk membuat sebuah tampilan halaman website yang dapat mempercepat pekerjaan seorang pengembang website ataupun pendesain halaman website. Sesuai namanya, website yang dibuat dengan alat bantu ini memiliki tampilan halaman yang sama / mirip dengan tampilan halaman Twitter atau desainer juga dapat mengubah tampilan halaman sesuai dengan ketentuan. Tampilan web yang dibuat bootstrap akan menyesuaikan ukuran layar *browser* yang kita gunakan baik di desktop, tablet ataupun *mobile device*. Fitur ini bisa diaktifkan ataupun dinon-aktifkan sesuai dengan keinginan kita sendiri. Sehingga, kita bisa membuat web untuk tampilan desktop saja dan apabila dirender oleh mobile browser maka tampilan dari web yang kita buat bisa beradaptasi sesuai layar. Dengan bootstrap kita juga bisa membangun web dinamis ataupun statis [15].

*Bootstrap* merupakan sebuah framework css yang memudahkan pengembang untuk membangun website yang menarik dan responsive. Tidak konsistensinya terhadap aplikasi individual membuat sulitnya untuk membangun dan memeliharanya. *Bootstrap* adalah css tetapi dibentuk dengan LESS, sebuah pre-processor yang member fleksibilitas dari css biasa. *Bootstrap* memberikan solusi dan seragam terhadap solusi yang umum, tugas *interface* yang setaip pengembang hadapi. Bootstrap dapat dikembangkan dengan tambahan lainnya karena ini cukup *flexible* terhadap pekerjaan design butuhkan [14].

Twitter *Bootstrap* adalah sebuah alat bantu atau framework atau bisa dibilang kerangka untuk membuat sebuah tampilan halaman website yang dapat mempercepat pekerjaan seorang pengembang website ataupun pendesain halaman website karena tidak dibutuhkan

*coding HTML, CSS*, maupun JavaScript terlalu banyak. Sesuai namanya, website yang dibuat dengan *bootstrap* memiliki tampilan halaman yang mirip dengan tampilan halaman Twitter. *Bootstrap* dibangun dengan teknologi HTML dan CSS yang dapat membuat layout halaman website, tabel, tombol, form, navigasi, dan komponen lainnya dalam sebuah website hanya dengan memanggil fungsi CSS (*class*) dalam berkas HTML yang telah didefinisikan. Selain itu juga terdapat komponen-komponen lainnya yang dibangun menggunakan JavaScript [14].

#### **a. Sejarah Bootstrap**

Istilah *bootstrap* berasal sejak awal abad ke-19 Amerika Serikat (khususnya dalam kalimat “menarik diri atas pagar atas *bootstrap* seseorang”) sepasang sepatu bot dengan satu *bootstrap* terlihat sepatu bot tinggi mungkin memiliki tab, lingkaran atau menangan di bagian atas yang dikenal sebagai *bootstrap*, yang memungkinkan seseorang untuk menggunakan jari atau alat *booting* kait untuk membantu menariknya. *Bootstrap* sudah digunakan selama abad ke-19” sebagai contoh tugas yang mustahil. Pada tahun 1834, ketika muncul di advokat workingman. Pada tahun 1860 muncul dalam komentar filsafat omneta *physical* yaitu upaya pemikiran yang menganalisis sendiri. *Bootstrap* sebagai metafora yang berarti yang berarti memperbaiki diri dengan upaya tanpa bantuan sendiri. Dan telah digunakan semenjak tahun 1922. *Bootstrap* istilah komputer mulai sebagai metafora pada 1950-an. Dalam komputer, menekan tombol *bootstrap* menyebabkan program tertanam untuk membaca program *bootstrap* dari unit input. komputer kemudian akan mengeksekusi program *bootstrap*, yang menyebabkan untuk membaca intruksi program yang lebih mendalam [15].

## **b. Pengembangan Bootstrap Dengan Perangkat Lunak**

Bootstrap juga dapat merujuk kepada pengembangan berturut-turut lebih kompleks, lingkungan pemograman lebih cepat. Lingkungan yang paling sederhana mungkin *editor* teks yang sangat dasar ( misalnya *red* ) dan program *assembler*. Menggunakan alat ini, seseorang dapat menulis *editor* teks yang lebih kompleks dan *compiler*.

Sederhana untuk bahasa tingkat tinggi dan seterusnya sampai seseorang memiliki grafis IDE dan bahasa pemograman tingkat tinggi yang sangat kuat. Secara historis, *bootstrap* juga mengacu pada teknik awal untuk pengembangan program komputer pada *hardware* baru. *Bootstrap* dalam pengembangan program dimulai pada 1950-an ketika setiap program dibangun diatas kertas dalam kode desimal atau dalam kode biner. Sedikit demi sedikit (1 0), karena tidak ada tingkat tinggi bahasa komputer, tidak ada *compiler*, *assembler* tidak ada, dan tidak ada *linker*. Sebuah program *assembler* kecil adalah tangan kode untuk komputer baru (misalnya IBM 650 ) yang di korvensi beberapa instruksi ke dalam lode biner atau desimal. *Compiler*, *linker*, *loader*, dan utilitas kemudian di kodekan dalam bahasa *assembly*, lanjut melanjut proses *bootstrap* pengembangan sistem *software* yang kompleks dengan menggunakan *software* sederhana. Istilah ini juga diperjuangkan oleh *Doug Engel bart* untuk merujuk keyakinannya bahwa organisasi bisa lebih baik berkembang dengan meningkatkan proses yang mereka gunakan untuk perbaikan ( sehingga mendapatkan efek peracikan dari waktu ke waktu ) tim SRI-nya yang mengembangkan *system hypertext* NLS meerapkan strategi ini dengan menggunakan alat yang telah dikembangkan.

### 2.2.12. UML




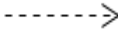



UML singkatan dari *Unified Modeling Language* yang berarti bahasa pemodelan standar. UML memiliki sintaks dan semantic. Ketika kita membuat model menggunakan konsep UML ada aturan – aturan yang harus diikuti. Bagaimana elemen pada model – model yang kita buat berhubungan satu dengan lainnya harus mengikuti standar yang ada. UML bukan hanya sekedar diagram, tetapi juga menceritakan konteksnya. Ketika pelanggan memesan sesuatu dari sistem, bagaimana transaksinya? Bagaimana sistem mengatasi error yang terjadi? Bagaimana keamanan sistem yang kita buat? Dan sebagainya dapat dijawab dengan UML [16]. Berikut ini adalah definisi mengenai 4 diagram UML [17].

#### *a. Use Case*



Use Case merupakan sebuah teknik yang digunakan dalam pengembangan sebuah software atau sistem informasi untuk menangkap kebutuhan fungsional dari sistem yang bersangkutan, Use Case menjelaskan interaksi yang terjadi antara ‘aktor’— inisiator dari interaksi sistem itu sendiri dengan sistem yang ada, sebuah Use Case direpresentasikan dengan urutan langkah yang sederhana.

Perilaku sistem adalah bagaimana sistem beraksi dan bereaksi. Perilaku ini merupakan aktifitas sistem yang bisa dilihat dari luar dan bisa diuji. Perilaku sistem ini dicapture di dalam USE CASE. USE CASE sendiri mendeskripsikan sistem, lingkungan sistem, serta hubungan antara sistem dengan lingkungannya.

Table 2.4 Simbol *Use Case* Diagram

Gambar	Nama	Keterangan
	<i>Actor</i>	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan <i>use case</i> .
	<i>Dependency</i>	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri ( <i>independent</i> ) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri ( <i>independent</i> ).
	<i>Generalization</i>	Hubungan dimana objek anak ( <i>descendent</i> ) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk ( <i>ancestor</i> ).
	<i>Include</i>	Menspesifikasikan bahwa <i>use case</i> sumber secara eksplisit.
	<i>Extend</i>	Menspesifikasikan bahwa <i>use case</i> target memperluas perilaku dari <i>use case</i> sumber pada suatu titik yang diberikan.
	<i>Association</i>	Apa yang menghubungkan antara objek satu dengan objek lainnya.
	<i>System</i>	Menspesifikasikan paket yang menampilkan sistem secara terbatas.



Gambar	Nama	Keterangan
	<i>Use Case</i>	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
	<i>Collaboration</i>	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).






#### **b. Activity Diagram**

Activity diagram adalah sesuatu yang menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity* diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. *Activity* diagram merupakan *state* diagram khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya atau internal *processing*.

Oleh karena itu *activity* diagram tidak menggambarkan *behaviour* internal sebuah sistem dan interaksi antar *subsistem* secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

Dan berikut adalah symbol dan contoh Activity Diagram

Table 2.5 Simbol *Activity Diagram*

Gambar	Nama	Keterangan
	<i>Activity</i>	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain.
	<i>Action</i>	State dari sistem yang mencerminkan eksekusi dari suatu aksi.
	<i>Initial Node</i>	Bagaimana objek dibentuk atau diawali.
	<i>Activity Final Node</i>	Bagaimana objek dibentuk dan dihancurkan
	<i>Fork Node</i>	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

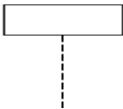


### c. *Sequence Diagram*

Diagram *sequence* merupakan salah satu yang menjelaskan bagaimana suatu operasi itu dilakukan; *message* (pesan) apa yang dikirim dan kapan pelaksanaannya. Diagram ini diatur berdasarkan waktu.

Obyek-obyek yang berkaitan dengan proses berjalannya operasi diurutkan dari kiri ke kanan berdasarkan waktu terjadinya dalam pesan yang terurut. Diagram *sequence* menampilkan interaksi antar objek dalam dua dimensi. Dimensi vertikal adalah poros waktu, dimana waktu berjalan ke arah bawah. Sedangkan dimensi horizontal merepresentasikan objek-objek individual. Tiap objek (termasuk *actor*) tersebut mempunyai waktu aktif yang direpresentasikan

dengan kolom vertical yang disebut dengan lifeline. Pesan (*message*) direpresentasikan sebagai panah dari satu lifeline ke lifeline yang lain. Message digambarkan sebagai garis berpanah dari satu objek ke objek lainnya. Pada fase desain berikutnya, message akan dipetakan menjadi operasi/metoda dari *class*.



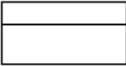


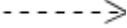

Table 2.6 Simbol *Sequence Diagram*

Gambar	Nama	Keterangan
	<i>LifeLine</i>	Objek <i>entity</i> , antarmuka yang saling berinteraksi.
	<i>Message</i>	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi.
	<i>Message</i>	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi.

#### **d. Class Diagram**

Class diagram adalah model statis yang menggambarkan struktur dan deskripsi class serta hubungannya antara class. Class diagram mirip ER-Diagram pada perancangan database, bedanya pada ER-diagram tdk terdapat operasi/methode tapi hanya atribut. Class terdiri dari nama kelas, atribut dan operasi/method [17].

Tabel 2.7 Simbol *Class Diagram*

Gambar	Nama	Keterangan
	<i>Generalization</i>	Hubungan dimana objek anak ( <i>descendent</i> ) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk ( <i>ancestor</i> ).
	<i>Nary Association</i>	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
	<i>Class</i>	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
	<i>Collaboration</i>	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor.
	<i>Realization</i>	Operasi yang benar-benar dilakukan oleh suatu objek.
	<i>Dependency</i>	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri ( <i>independent</i> ) akan memengaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
	<i>Association</i>	Apa yang menghubungkan antara objek satu dengan objek lainnya.

### 2.2.13. Dokumen PDF

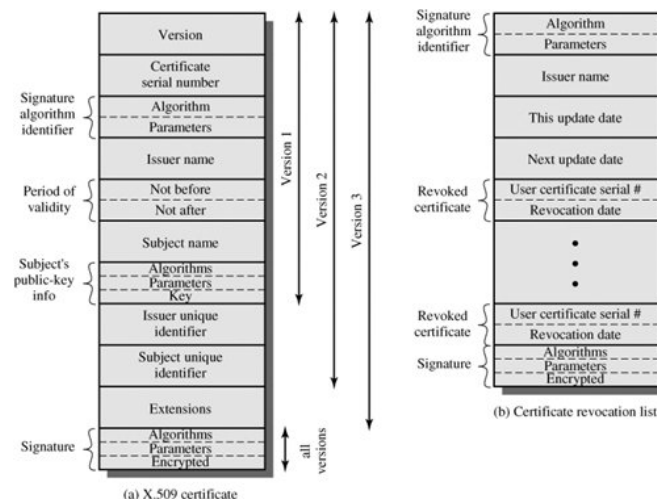
Portable Document Format (disingkat PDF) adalah sebuah format berkas yang dibuat oleh Adobe Systems pada tahun 1993 untuk keperluan pertukaran dokumen digital. Format PDF digunakan untuk merepresentasikan dokumen dua dimensi yang meliputi teks, huruf, citra dan grafik vektor dua dimensi. Pada Acrobat 3-D, kemampuan PDF juga meliputi pembacaan dokumen tiga dimensi. PDF telah menjadi standar ISO pada tanggal 1 Juli 2008 dengan kode ISO 32000-1:2008 [19].

### 2.2.14. Sertifikat X.509

Standard untuk sertifikat telah ditetapkan dan disetujui oleh ITU. Standard tersebut dinamakan X.509 dan digunakan secara luas di internet. Ada tiga versi standard X.509, yaitu V1, V2, dan V3 [21]. Inti dari skema X.509 adalah sertifikat kunci publik yang diasosiasikan pada setiap pengguna. Pengguna dari sertifikat ini diasumsikan membuat sertifikat dari suatu otoritas penerbit sertifikat (*Certificate Authority/ CA*). Direktori *server* menyediakan tempat yang mudah diakses oleh pengguna untuk mendapatkan sertifikat. Format umum dari sertifikat seperti terlihat pada Gambar 3), di antaranya[20]:

- *Version*: Pada umumnya sertifikat adalah versi 1. Jika *Issuer Unique Identifier* atau *Subject Unique Identifier* terdapat dalam sertifikat, maka versi sertifikat tersebut adalah versi 2. Jika ada satu atau lebih tambahan, maka versi sertifikat tersebut adalah versi 3.
- *Serial number*: sebuah nilai bilangan bulat, yang unik yang dikeluarkan oleh CA.
- *Signature algorithm identifier*: algoritma yang digunakan untuk menandatangani sertifikat, beserta parameter yang terkandung dalam sertifikat.

- *Issuer name*: nama X.500 dari CA yang membuat dan menandatangani sertifikat.
- *Period of validity*: Terdiri dari dua tanggal, masa pembuatan dan kadaluarsa dari sertifikat.
- *Subject name*: nama dari pengguna yang membuat sertifikat. Sertifikat ini menjamin kunci publik dari pemegang kunci privat.
- *Subject's public key information*: kunci publik dari subjek ditambah dengan identifikasi dari algoritma yang digunakan.
- *Issuer unique identifier*: pilihan tambahan yang digunakan sebagai identifikasi secara unik mengenai CA.
- *Subject unique identifier*: pilihan tambahan yang digunakan sebagai identifikasi secara unik mengenai pemegang sertifikat.
- *Extensions*: kumpulan dari satu atau lebih bagian tambahan. Tambahan ini terdapat pada sertifikat versi 3.
- *Signature*: meliputi seluruh bagian dari sertifikat; yang berisi kode hash yang dienkripsi oleh kunci privat milik CA. Pada bagian ini juga terdapat identifikasi algoritma signature.



Gambar 2.5 Format X.509

Sertifikat pengguna yang dibuat oleh CA memiliki karakteristik sebagai berikut:

- Setiap pengguna dengan akses ke kunci publik CA dapat memverifikasi kunci publik pengguna yang telah disertifikasi.
- Tidak ada pihak selain otoritas sertifikasi yang dapat mengubah sertifikat tanpa terdeteksi.

Karena sertifikat tidak dapat dipalsukan, sertifikat dapat ditempatkan di direktori tanpa perlu melakukan upaya khusus untuk melindunginya [20].

Jika semua pengguna berlangganan CA yang sama, maka ada kepercayaan umum dari CA itu. Semua sertifikat pengguna dapat ditempatkan di direktori untuk diakses oleh semua pengguna. Selain itu, pengguna dapat mengirimkan sertifikatnya langsung ke pengguna lain. Dalam kedua kasus, setelah B memiliki sertifikat A, B memiliki keyakinan bahwa pesan yang dienkripsi dengan kunci publik A akan aman dari penyadapan dan pesan yang ditandatangani dengan kunci pribadi A tidak dapat dipalsukan [20].

Jika ada komunitas pengguna yang besar, mungkin tidak praktis bagi semua pengguna untuk berlangganan CA yang sama. Karena CA yang menandatangani sertifikat, setiap pengguna yang berpartisipasi harus memiliki salinan kunci publik CA sendiri untuk memverifikasi tanda tangan. Kunci publik ini harus diberikan kepada setiap pengguna dengan cara yang benar-benar aman (berkenaan dengan integritas dan keaslian) sehingga pengguna memiliki kepercayaan pada sertifikat terkait. Jadi, dengan banyak pengguna, mungkin lebih praktis jika ada sejumlah CA, yang masing-masing menyediakan kunci publiknya dengan aman ke sebagian kecil pengguna.

Sekarang anggaplah A telah memperoleh sertifikat dari otoritas sertifikasi X1 dan B telah memperoleh sertifikat dari CA X2. Jika A tidak mengetahui kunci publik X2 dengan aman, maka sertifikat B, yang dikeluarkan oleh X2, tidak berguna bagi A. A dapat membaca sertifikat B, tetapi A tidak dapat memverifikasi tanda tangannya. Namun, jika dua CA telah secara aman menukar kunci publik mereka

sendiri, prosedur berikut akan memungkinkan A untuk mendapatkan kunci publik B:

- 1) A memperoleh, dari direktori, sertifikat X2 yang ditandatangani oleh X1. Karena A mengetahui kunci publik X1 dengan aman, A dapat memperoleh kunci publik X2 dari sertifikatnya dan memverifikasinya melalui tanda tangan X1 pada sertifikat.
- 2) A kemudian kembali ke direktori dan mendapatkan sertifikat B yang ditandatangani oleh X2. Karena A sekarang memiliki salinan kunci publik X2 yang tepercaya, A dapat memverifikasi tanda tangan dan mendapatkan kunci publik B dengan aman.

A telah menggunakan rantai sertifikat untuk mendapatkan kunci publik B. Dalam notasi X.509, rantai ini dinyatakan sebagai

$X1 \ll X2 \gg X2 \ll B \gg$

Dengan cara yang sama, B dapat memperoleh kunci publik A dengan rantai terbalik:

$X2 \ll X1 \gg X1 \ll A \gg$

Skema ini tidak perlu dibatasi pada rantai dua sertifikat. Jalur CA yang panjang dan sewenang-wenang dapat diikuti untuk menghasilkan rantai. Sebuah rantai dengan elemen N akan dinyatakan sebagai

$X1 \ll X2 \gg X2 \ll X3 \gg \dots XN \ll B \gg$

Dalam hal ini, setiap pasangan CA dalam rantai  $(X_i, X_{i+1})$  harus telah membuat sertifikat satu sama lain.

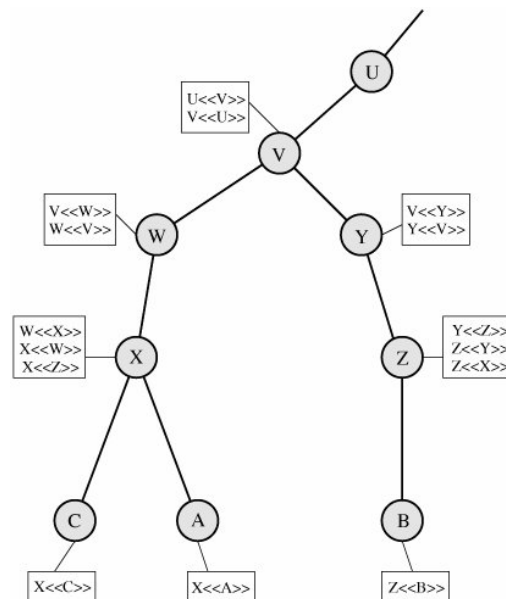
Semua sertifikat CA oleh CA ini perlu muncul di direktori, dan pengguna perlu mengetahui bagaimana mereka ditautkan untuk mengikuti jalur ke sertifikat kunci publik pengguna lain. X.509 menyarankan agar CA diatur dalam hierarki sehingga navigasi menjadi mudah.

Gambar 2.6, diambil dari X.509, adalah contoh hierarki tersebut. Lingkaran yang terhubung menunjukkan hubungan hierarkis di antara CA; kotak terkait menunjukkan sertifikat yang disimpan dalam



direktori untuk setiap entri CA. Entri direktori untuk setiap CA mencakup dua jenis sertifikat:

- Teruskan sertifikat: Sertifikat X yang dihasilkan oleh CA lain
- Sertifikat terbalik: Sertifikat yang dihasilkan oleh X yang merupakan sertifikat CA lain



Gambar 2.6 Hirarkhi CA di dalam PKI

Dalam contoh ini, pengguna A dapat memperoleh sertifikat berikut dari direktori untuk menetapkan jalur sertifikasi ke B:

$X \ll W \gg W \ll V \gg V \ll Y \gg Y \ll Z \gg Z \ll B \gg$

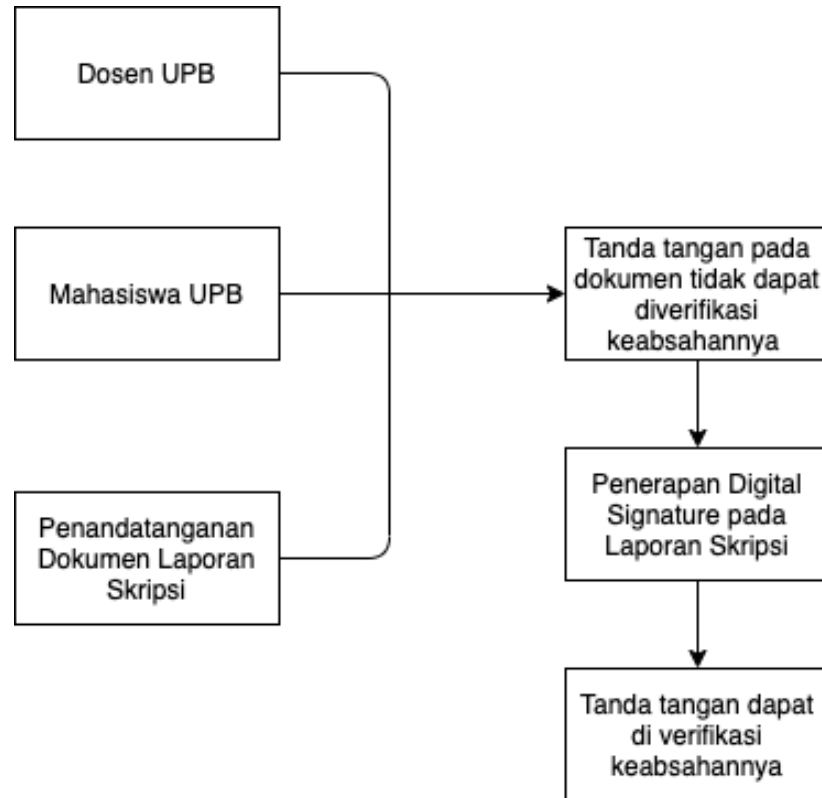
Saat A memperoleh sertifikat ini, A dapat membuka jalur sertifikasi secara berurutan untuk memulihkan salinan kunci publik B yang tepercaya. Dengan menggunakan kunci publik ini, A dapat mengirim pesan terenkripsi ke B. Jika A ingin menerima pesan terenkripsi kembali dari B, atau menandatangani pesan yang dikirim ke B, maka B akan memerlukan kunci publik A, yang dapat diperoleh dari jalur sertifikasi berikut:

$Z \ll Y \gg Y \ll V \gg V \ll W \gg W \ll X \gg X \ll A \gg$

B dapat memperoleh set sertifikat ini dari direktori, atau A dapat memberikannya sebagai bagian dari pesan awalnya kepada B.

### 2.3. Kerangka Berfikir

Berikut kerangka pemikiran penelitian yang dilakukan oleh penulis :



Gambar 2.7 kerangka berfikir

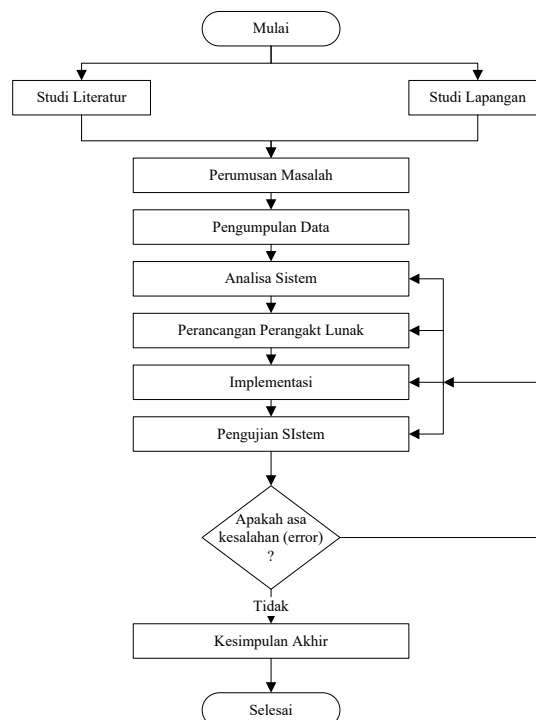
## BAB III

### METODE PENELITIAN

Pada dasarnya suatu penelitian bertujuan untuk menemukan, mengembangkan atau mengkaji suatu pengetahuan. Menemukan dapat diartikan sebagai usaha untuk mendapatkan sesuatu, dalam usaha untuk mengisi kekosongan atau kekurangan. Metodologi penelitian merupakan suatu cara atau tindakan peneliti dalam pencarian data dan menggunakan data tersebut untuk dapat dijadikan sebagai sebuah informasi yang lebih akurat. Dengan metodologi penelitian yang jelas akan lebih mempermudah alur penelitian, sehingga dapat menghasilkan penelitian yang maksimal. Metode penelitian ini digunakan sebagai pedoman penelitian dalam pelaksanaan penelitian agar hasil yang dicapai tidak menyimpang dari tujuan yang telah ditentukan sebelumnya.

#### 3.1. Tahapan Penelitian

Tahapan-tahapan yang akan dilakukan dalam metodologi penelitian dapat dilihat pada gambar 3.1.



Gambar 3.1 Tahapan Penelitian

### **3.2. Studi Literatur**

Pada tahap ini penulis melakukan studi terhadap beberapa alat bantu dan konsep yang akan digunakan dalam penelitian ini. Studi dilakukan pada beberapa alat bantu yang akan digunakan untuk membangun sistem dalam penelitian ini.

Studi juga dilakukan dengan mempelajari berbagai macam buku teks, diktat kuliah, jurnal, karya tulis ilmiah, tugas akhir dan tesis yang berkaitan dengan masalah yang akan dibahas yaitu kriptografi khususnya metode kriptografi RSA dan *digital signature*, sehingga penulis mendapatkan dasar-dasar referensi yang kuat dalam menentukan metode yang tepat untuk menyelesaikan permasalahan yang akan diteliti.

### **3.3. Studi Lapangan**

Pada tahap ini penulis melakukan studi kasus dalam melakukan penandatanganan dokumen laporan skripsi pada Universitas Pelita Bangsa untuk dapat mengetahui permasalahan-permasalahan yang ada, yang kemudian akan dirumuskan saat melakukan perumusan masalah.

### **3.4. Perumusan Masalah**

Pada tahap ini penulis melakukan penentuan masalah yang akan diselesaikan dalam tugas akhir ini, yaitu implementasi atau penerapan *digital signature* dengan algoritma RSA untuk mengetahui keabsahan dokumen laporan skripsi.

### **3.5. Pengumpulan Data**

Pada tahap ini penulis melakukan pengumpulan data tentang aplikasi dan data terkait laporan skripsi. Penulis menggunakan beberapa metode pengumpulan data, yaitu:

#### **1. Pengamatan (*Observation*)**

Metode obvasi dalam sejarah perkembangan ilmu pengetahuan bersumber dari dunia emiris, sejak observasi botanis Aristoteles hingga

observasi historis Herodotus tentu berdasarkan pada kehidupan, penggambaran, dan pengalaman langsung. Sedangkan Auguste Comte (perintis ilmu sosiologi, mengukuhkan bahwa observasi merupakan satu diantara empat metode penelitian yang banyak digunakan oleh para peneliti, sesuai dengan embrio ilmu pengetahuan social. Mendatangi langsung ke lokasi kegiatan dengan melihat, dan mengamati kegiatan-kegiatan yang dilakukan pada Universitas Pelita Bangsa dalam melakukan proses penandatanganan dokumen skripsi yang sedang berlangsung.

## 2. Wawancara

Wawancara adalah metode pengumpulan data dengan tanya jawab secara langsung kepada dosen yang memiliki pemahaman lebih dalam hal yang sedang diteliti.

### 3.6. Identifikasi Masalah

Setelah melakukan pengumpulan data, langkah berikutnya adalah mengidentifikasi permasalahan sistem yang akan dibuat sesuai dengan batasan yang ada. Dalam identifikasi permasalahan ini, analisa yang dibutuhkan dalam penyelesaian masalah penentuan keabsahan dokumen laporan skripsi ini akan dilakukan dalam beberapa tahapan antara lain :

#### 3.6.1. Analisis Data

Analisa data merupakan salah satu tahapan yang penulis lakukan untuk menyelesaikan permasalahan yang diteliti, didalam analisa data ini penulis melakukan :

- a. Pengumpulan data yang berfungsi untuk memperoleh data yang penulis perlukan dalam perancangan program.
- b. Penulis melakukan pengelompokan data sesuai dengan jenis dan fungsinya.

- c. Pendeskripsian data untuk menentukan langkah-langkah yang harus digunakan untuk membangun sistem yang mudah digunakan dan dengan tampilan yang menarik bagi penggunaanya.

### **3.6.2. Analisa Penerapan Algoritma**

Tahapan selanjutnya adalah analisa penerapan algoritma yang dilakukan setelah pengumpulan data. Analisa penerapan algoritma menjelaskan tahapan untuk menerapkan *digital signature* dengan algoritma asimetris RSA pada dokumen laporan skripsi untuk menentukan keabsahan dokumen tersebut. Tahapan-tahapan yang dilakukan adalah :

- a. Proses pembuatan sertifikat digital x.509 yang digunakan untuk proses penandatanganan dokumen laporan skripsi.
- b. Proses pembuatan *field* tanda tangan digital pada dokumen pdf.
- c. Proses penandatanganan dokumen laporan skripsi.
- d. Proses verifikasi tanda tangan pada dokumen laporan skripsi.

### **3.6.3. Analisa Sistem**

Implementasi *digital signature* dengan algoritma RSA ini akan digunakan untuk menentukan keabsahan dokumen laporan skripsi. Oleh karenanya dibutuhkan suatu sistem yang dapat melakukan proses enkripsi dan proses verifikasi terhadap dokumen yang telah ditandatangani secara digital.

Sistem yang dikembangkan berupa aplikasi *website* yang dapat digunakan oleh mahasiswa dan dosen Universitas Pelita bangsa dalam melakukan aktifitas penandatanganan dokumen laporan skripsi. Menggantikan metode penandatanganan dengan menempelkan hasil scan tanda tangan ke dalam dokumen laporan skripsi.

Dalam analisa sistem ini penulis akan menggunakan beberapa alat bantu diantaranya *use case* diagram, *activity* diagram dan *sequence* diagram.

### 3.7. Perancangan Perangkat Lunak

Pada tahap ini penulis melakukan perancangan sesuai dengan hasil analisa sistem khususnya perancangan modul-modul yang digunakan untuk penandatanganan digital dan verifikasi dokumen, dan modul-modul pendukung lainnya serta. Dalam pengembangan sistem akan digunakan metode *scrum*.

### 3.8. Implementasi

Pada proses implementasi ini penulis melakukan pembuatan modul-modul yang telah dirancang dalam tahap perancangan kedalam bahasa pemrograman *python*.

Berikut instrumen yang digunakan penulis untuk membangun sistem dalam penelitian ini :

Tabel 3.1 Daftar *Software* yang digunakan

No	Nama	Versi	Keterangan
1.	Sistem Operasi MacOS	10.15.7 (Catalina)	Sistem operasi yang di keluarkan oleh perusahaan Apple.
2.	Visual Studio Code	1.57.1	<i>SoftwareIDE</i> yang digunakan untuk text editing dalam melakukan coding aplikasi.
3.	MAMP	6.3	Merupakan aplikasi yang digunakan untuk menjalankan database MySQL. Selain itu juga dapat digunakan untuk menjalankan server PHP.

4.	Chrome	91.0.4472.114	Browser yang dibuat oleh perusahaan google digunakan untuk testing website dan inspeksi element HTML.
----	--------	---------------	---

Tabel 3.2 Daftar *Hardware* yang digunakan

No	Nama	Spesifikasi	
1.	CPU	Merk	Dell
		Tipe	Optiplex 3020M
		Ram	10 Gb
		Prosesor	Intel Core i3 4000 series
		Penyimpanan	SSD 256 Gb

### 3.9. Pengujian Sistem

Tahap pengujian dilakukan dengan tujuan untuk menjamin bahwa sistem yang dibangun sesuai dengan hasil analisis dan perancangan serta menghasilkan suatu kesimpulan apakah sistem tersebut sesuai dengan yang diharapkan. Pengujian dilakukan oleh user yang bersangkutan dalam pengoperasian sistem nantinya. Pengujian merujuk pada setiap fungsi dan fitur yang sebelumnya telah ditentukan pada pengumpulan data. Adapun metode yang digunakan pada pengujian ini yaitu metode *Blackbox*. Pengujian *blackbox* (*blackbox testing*) adalah salah satu metode pengujian perangkat lunak yang berfokus pada sisi fungsionalitas, khususnya pada input dan output aplikasi (apakah sudah sesuai dengan apa yang diharapkan atau belum). Tahap pengujian atau testing merupakan salah satu tahap yang harus ada dalam sebuah siklus pengembangan perangkat lunak (selain tahap perancangan atau desain).

### 3.10. Kesimpulan Akhir

Pada tahapan ini diambil kesimpulan akhir dalam penerapan dari *digital signature* dengan algoritma RSA untuk menentukan keabsahan dokumen laporan skripsi, berdasarkan hasil pengujian yang telah dilakukan, untuk



mengetahui apakah implementasi metode *digital signature* yang telah dilakukan dapat membuktikan keabsahan dokumen laporan skripsi. Dan tidak lupa juga pada tahap ini diberikan saran untuk perbaikan pengembangan sistem.

## **BAB IV**

### **HASIL PENELITIAN DAN PEMBAHASAN**

#### **4.1. Pengumpulan Data**

Pada tahap ini penulis melakukan proses pengumpulan bahan penelitian berupa objek, yang menjadi objek dalam penelitian ini adalah data elektronik/*digital*. Data *digital* yang dimaksud adalah file dokumen laporan skripsi berekstensi *pdf*.

#### **4.2. Analisis Sistem**

##### **4.2.1. Analisis Masalah**

Dokumen Laporan Skripsi merupakan salah satu hal penting dalam menyelesaikan perkuliahan jenjang S1. Di era digitalisasi seperti saat ini laporan skripsi disusun dalam bentuk file pdf. Dalam penyusunannya terdapat beberapa tahapan-tahapan yang harus dilalui antara lain mulai dari penyusunan laporan disertai dengan bimbingan dengan dosen pembimbing yang telah ditetapkan sebelumnya. Ketika laporan Skripsi telah selesai disusun kemudian dilanjutkan dengan menandatangani lembar persetujuan oleh dosen yang bersangkutan dan tidak lupa penandatanganan lampiran yang mengharuskan untuk ditandatangani. Setelah ditanda tangani selanjutnya mahasiswa dapat melakukan sidang skripsi. Setelah melakukan sidang skripsi maka ketika ada revisi terkait laporan tersebut harus dikerjakan dan kemudian setelahnya melanjutkan ke tahap berikutnya yaitu penandatanganan Lembar Pengesahan Laporan Skripsi.

Ketika lembar pengesahan telah selesai di tanda tangani maka laporan skripsi dianggap telah selesai disusun. Masalah muncul ketika dosen yang menandatangani dokumen laporan dan pemilik laporan tersebut tidak dapat memverifikasi tanda tangan yang tertera pada laporan skripsi. Dengan demikian dapat dengan mudah tanda tangan tersebut di manipulasi. Selama ini penandatanganan dokumen

Laporan skripsi pada Universitas Pelita Bangsa adalah dengan menempelkan gambar tanda tangan ke dokumen yang ada. Hal ini menyebabkan tanda tangan dapat dengan mudah untuk ditiru.

Untuk mengatasi kekurangan tersebut maka perlu dibuatkan suatu sistem yang dapat menggantikan metode penandatanganan dokumen skripsi tersebut. Sistem yang dibuat dapat digunakan oleh mahasiswa untuk melakukan permintaan tanda tangan dan dapat digunakan oleh dosen untuk menandatangani dokumen. Metode penandatanganan yang baru adalah dengan memanfaatkan tanda tangan digital.

Tanda tangan digital merupakan salah satu implementasi kriptografi asimetris. Cara kerja dan kegunaan tanda tangan digital mirip dengan tanda tangan versi nyata, yaitu untuk memberikan kepastian keaslian dan persetujuan dokumen oleh penanda tangan. Dengan adanya sistem ini maka diharapkan dapat mengganti metode penanda tangan dokumen digital sebelumnya. Dan dapat memberikan kepercayaan bagi pemilik dokumen bahwa penanda tangan dokumen tersebut adalah orang yang tepat.

#### **4.2.2. Analisis Kebutuhan Sistem**

Pada tahap ini penulis akan memulai mengumpulkan kebutuhan dalam melakukan penelitian seperti analisis kebutuhan *input* dan *output*, serta analisa kebutuhan perangkat keras dan perangkat lunak yang akan digunakan dalam perancangan sistem.

##### **a. Analisis *Input* Sistem**

Pada penelitian ini *input* sistem yang dibangun adalah *file* dari data elektornik/*digital* dengan ekstensi PDF. Dokumen PDF memiliki struktur sebagai berikut :

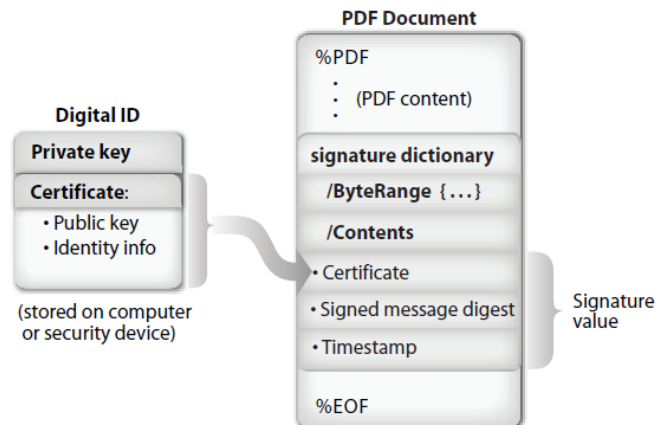


demikian, penampil PDF tidak perlu memindai seluruh file untuk menemukan objek.

- **Trailer** – menentukan lokasi Tabel Xref dan objek lainnya.
- Format PDF dirancang untuk dibaca dari akhir, untuk menemukan Tabel Xref dengan cepat. Baris terakhir dokumen harus berisi penanda %%EOF (End-of-File).

#### b. Analisis *Output* Sistem

Penelitian ini menghasilkan *output* berupa *file* PDF yang telah ditanda tangani. PDF menyertakan dukungan untuk tanda tangan untuk disematkan dalam dokumen itu sendiri, daripada dikelola sebagai data terpisah atau ditambahkan ke format dokumen yang ada. Setiap tanda tangan digital dalam dokumen PDF dikaitkan dengan penanganan tanda tangan. Tanda tangan ditempatkan dalam kamus tanda tangan PDF yang berisi nama handler tanda tangan yang akan digunakan untuk memproses tanda tangan tersebut (Gambar 4.2). Pengendali tanda tangan yang ada di dalam Adobe Acrobat memanfaatkan teknologi kriptografi Kunci Publik/Privat (PPK). PPK didasarkan pada gagasan bahwa nilai yang dienkripsi dengan kunci pribadi hanya dapat didekripsi menggunakan kunci publik. Saat PDF ditandatangani, sertifikat penandatanganan disematkan dalam file PDF. Gambar 4.1 menunjukkan hubungan antara ID digital yang disimpan pada perangkat keras pengguna dan nilai tanda tangan yang disematkan dalam dokumen PDF. Nilai tanda tangan juga dapat mencakup informasi tambahan seperti grafik tanda tangan, cap waktu, dan data lain yang mungkin spesifik untuk pengguna, sistem, atau aplikasi.

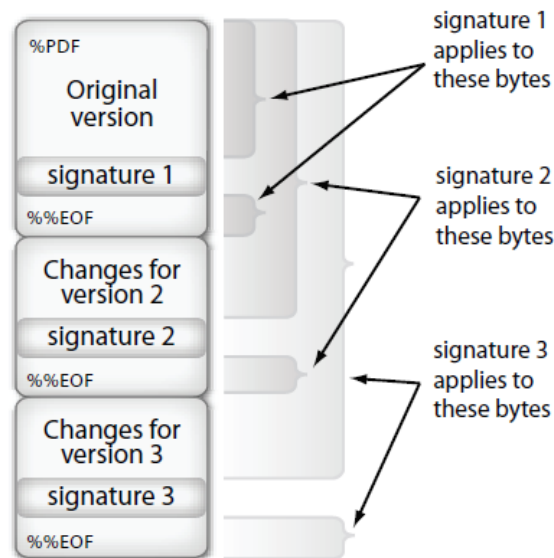


Gambar 4.2. Digital ID dan dokumen PDF yang telah ditandatangani

Dokumen PDF Laporan Skripsi pada lembar pengesahannya membutuhkan beberapa tanda tangan. Sangat mudah untuk menanganinya dengan dokumen kertas hanya dengan menggambar garis lain di atas kertas. Di dunia kertas, seseorang yang menandatangani dokumen akan bijaksana untuk menyimpan salinan dokumen seperti yang ditandatangani. Kemudian jika orang lain mengubah dokumen, penandatanganan dapat dengan mudah berargumen bahwa dokumen tersebut telah diubah. Namun, dengan PDF, setiap upaya untuk mengubah dokumen dengan memodifikasi file (seperti menandatangani lagi) akan membatalkan tanda tangan digital yang ada. Hal ini terjadi karena nilai hash yang dihitung pada waktu verifikasi tidak akan cocok dengan hash terenkripsi yang dibuat pada waktu penandatanganan. PDF memecahkan masalah ini dengan mendukung kemampuan untuk melakukan pembaruan tambahan, penandatanganan dapat menambahkan bidang tanda tangan lain ke dokumen dan menandatangani tanpa membatalkan tanda tangan sebelumnya.

Format file PDF mendefinisikan kemampuan pembaruan tambahan. Pembaruan tambahan transparan bagi orang yang melihat dokumen, tetapi memungkinkan deteksi dan audit

modifikasi pada file. Fitur bahasa PDF ini secara umum, dan file PDF yang ditandatangani secara khusus, memungkinkan file PDF apa pun untuk dimodifikasi dengan menambahkan informasi modifikasi ke akhir file di bagian pembaruan tambahan. Tidak ada perubahan apa pun yang diperlukan untuk byte yang mewakili versi file sebelumnya. Ini memungkinkan tanda tangan tambahan ditambahkan ke file PDF tanpa mengubah data apa pun yang dicakup oleh tanda tangan sebelumnya. Setiap tanda tangan tambahan akan mencakup seluruh file PDF, dari byte 0 hingga byte terakhir, tidak termasuk hanya nilai tanda tangan untuk nilai tanda tangan saat ini. Gambar 4.3 menunjukkan bagaimana tanda tangan dibuat untuk file dengan tiga tanda tangan.



Gambar 4.3. *Multiple Signature* pada dokumen PDF

### c. Analisis Kebutuhan *Hardware* dan *Software*

Dalam melakukan penelitian ini penulis memerlukan beberapa alat bantu diantaranya adalah : untuk *software* penulis menggunakan MAMP untuk server database, Browser Chrome untuk melakukan testing sistem yang dibangun dan Visual Studio Code sebagai IDE untuk menulis coding. Sedangkan untuk

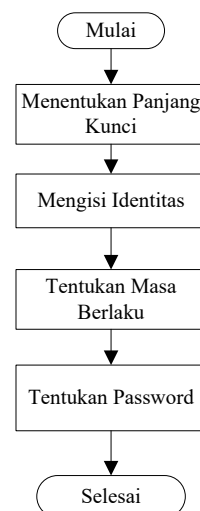
*hardware* penulis menggunakan CPU dengan spesifikasi Intel Core i3 4000 series, 10Gb RAM, dan 256Gb SSD, sistem operasi menggunakan MacOS Catalina.

#### 4.2.3. Analisis Proses

Tahap ini penulis melakukan analisis terhadap proses terkait penerapan tanda tangan digital, antara lain :

a. Proses Pembuatan Sertifikat Digital X.509

Proses pembuatan sertifikat digital X.509 ditunjukkan pada Gambar 4.4. pada Gambar 4.4 proses pembuatan sertifikat. Proses Diawali dengan menentukan panjang kunci sebesar 4096 *bit*. Setelah menentukan panjang kunci, proses berlanjut dengan mengisi data pemilik sertifikat. Proses selanjutnya adalah menentukan masa berlaku sertifikat dan password sertifikat. Proses ini menghasilkan sertifikat digital X.509 yang tertera informasi pemiliknya disertai dengan kunci publiknya dan menghasilkan kunci pribadi. Keduanya disimpan pada sistem yang akan dibuat untuk nantinya digunakan dalam penandatanganan dokumen laporan skripsi.

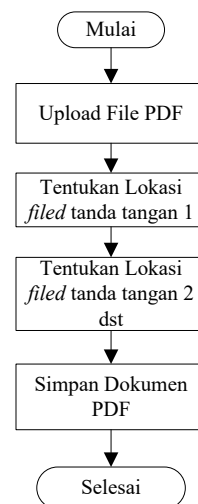


Gambar 4.4 Proses Pembuatan Sertifikat digital X.509

b. Proses pembuatan *field* tanda tangan.



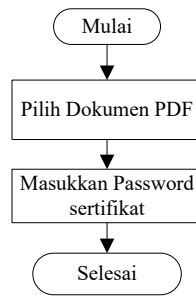
Proses pembuatan *field* tanda tangan digital ditunjukkan pada Gambar 4.5. Pada proses ini diawali dengan mengunggah dokumen skripsi yang akan diberi tanda tangan. Tahap selanjutnya adalah penentuan lokasi *field* pada dokumen yang telah diunggah. Penempatan lokasi ini dilakukan sebanyak tanda tangan yang diperlukan pada dokumen tersebut. Dan tahap akhir adalah penyimpanan dokumen yang telah ditambahkan *field* tanda tangan didalamnya. Proses ini menghasilkan dokumen skripsi yang memiliki *field* tanda tangan didalamnya yang selanjutnya siap untuk ditandatangani.



Gambar 4.5 Proses Pembuatan *field* Tanda Tangan

c. Proses Penandatanganan Dokumen

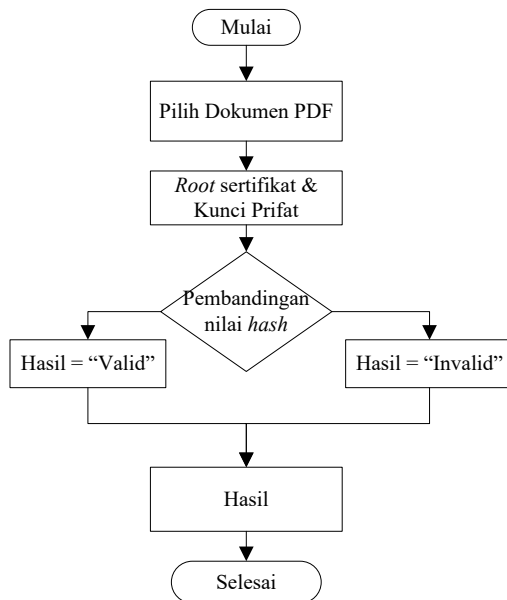
Proses penandatanganan dokumen skripsi ditunjukkan pada Gambar 4.6. Pada proses ini diawali dengan dosen memilih dokumen yang akan ditandatangani yang sebelumnya telah diunggah oleh mahasiswa. Setelah dokumen dipilih dosen dapat melakukan penandatanganan dan kemudian mengisi *password* sertifikat yang telah dimiliki sebelumnya. Proses ini menghasilkan dokumen laporan skripsi yang telah ditandatangani.



Gambar 4.6 Proses Penandatanganan Dokumen

d. Proses Verifikasi

Proses verifikasi tanda tangan digital ini ditunjukkan pada Gambar 4.7. Proses diawali dengan pemilihan dokumen yang akan diverifikasi. Setelah dokumen dipilih dengan menggunakan kunci privat serta sertifikat *root* akan dilakukan verifikasi dengan cara membandingkan *hash* yang ada pada dokumen dan jika hasilnya sama maka tanda tangan *valid* dan jika berbeda maka hasilnya *invalid*.



Gambar 4.7. Proses Verifikasi

### 4.3. Perancangan

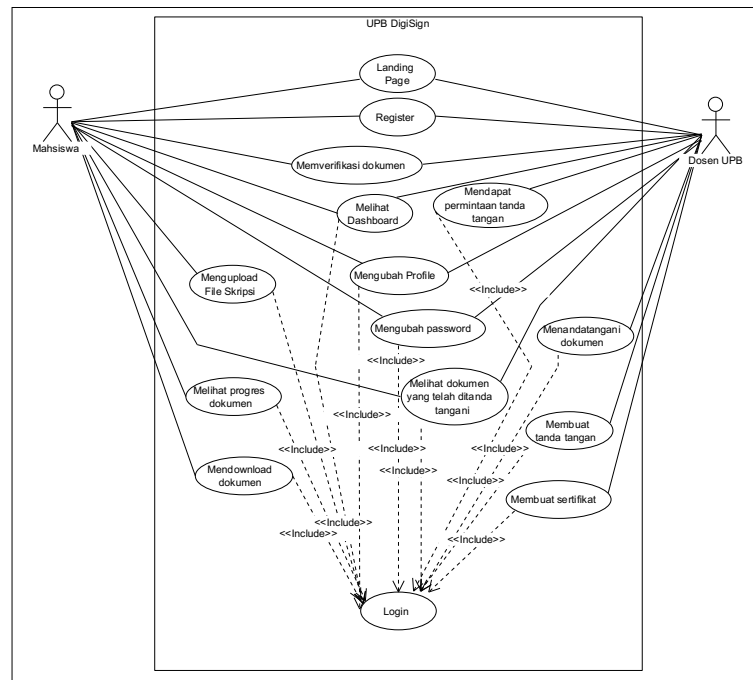
Tahapan ini merupakan perancangan sistem, perancangan sistem merupakan spesifikasi dari *solution* berbasis komputer secara *detail*, disebut juga dengan *physical design*. Perancangan sistem menekankan pada implementasi sistem secara teknis (Whitten & Bentley 2007).

Perancangan sistem bertujuan untuk menampilkan gambaran sistem yang akan dibangun sehingga mempermudah penulis dalam melakukan implementasi maupun evaluasi.

Pada tahap perancangan sistem ini, penulis akan membuat beberapa rancangan sistem antara lain *use case* diagram, *activity* diagram dan *sequence* diagram untuk nantinya akan dibuat pada tahap implementasi sistem.

#### 4.3.1. Usecase Diagram

*Use case* diagram menggambarkan bagaimana proses yang berlangsung pada sistem dan tidak ketinggalan aktor yang berperan dalam sistem tersebut. Ada 2 aktor yang terlibat dalam sistem yang akan dibuat. Pada gambar dibawah ini merupakan *use case* diagram sistem.



Gambar 4. *Usecase* diagram sistem

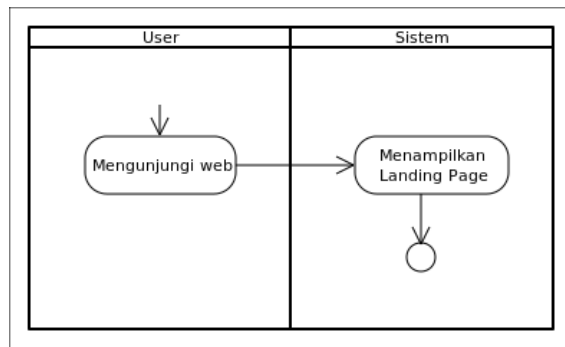
Berdasarkan gambar *use case* diagram diatas terdapat :

- a. Satu sistem yang mencakup kegiatan penandatanganan dokumen laporan skripsi
- b. Aktor yang dapat mengakses sistem, aktor yang dimaksud antar lain:
  - 1) Mahasiswa, sebagai pengguna yang akan melakukan permintaan penandatanganan
  - 2) Dosen, sebagai pengguna sistem yang melakukan penandatanganan dokumen

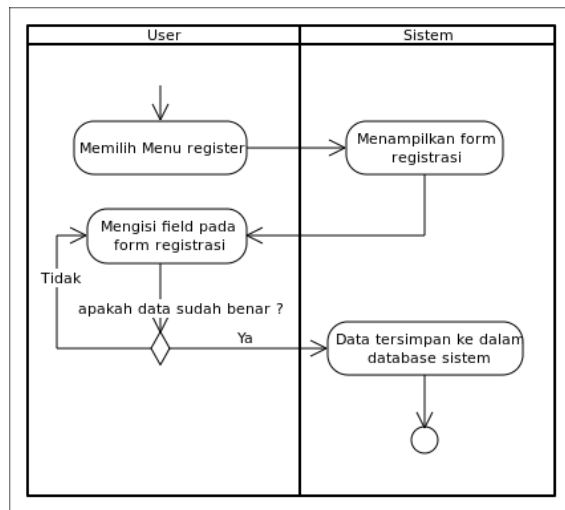
#### 4.3.2. Activity Diagram

Dari *use case* diatas didapat beberapa *Activity Diagram* yang akan penulis jelaskan sesuai dengan fungsi *Activity Diagram* yaitu untuk menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum. *Activity Diagram* tersebut antara lain :

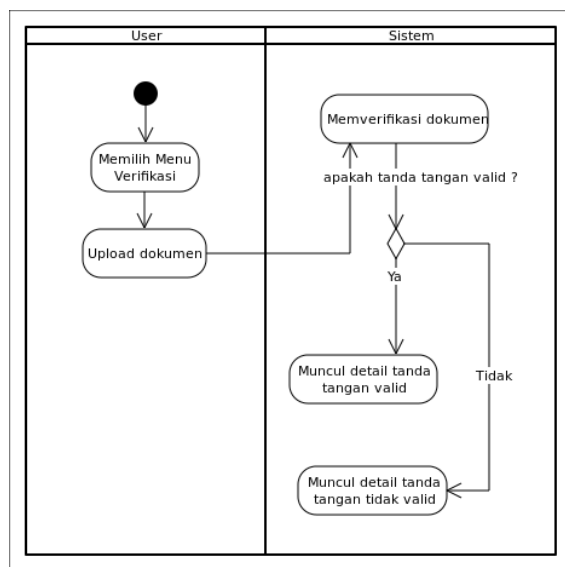
- a. *Activity Diagram Landing Page*



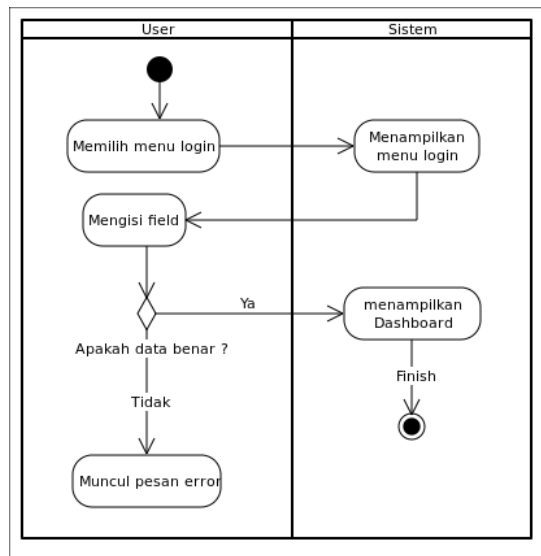
b. *Activity Diagram Register Page*



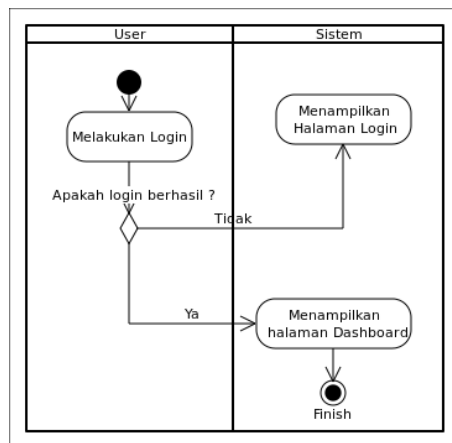
c. *Activity Diagram Verifikasi Dokumen*



d. *Activity Diagram Login Page*



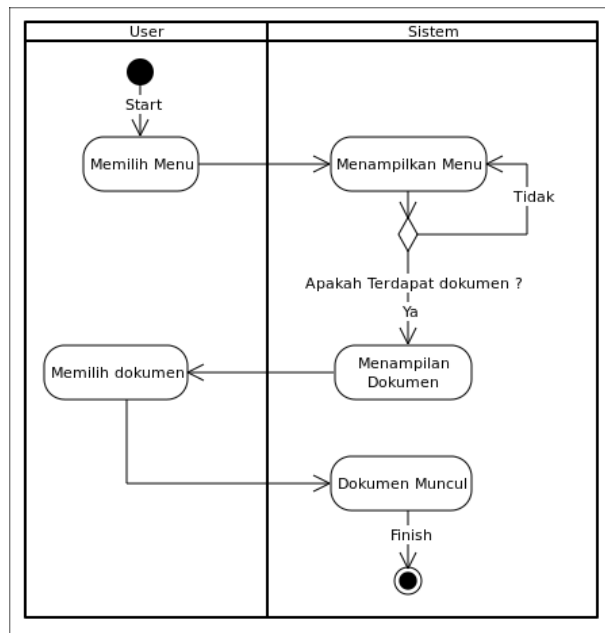
e. *Activity Diagram Dashboard*



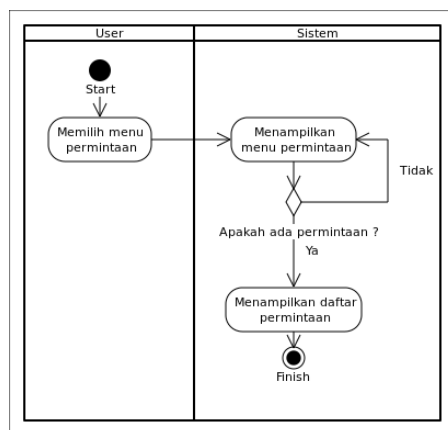
f. *Activity Diagram Mengubah Profile*

g. *Activity Diagram Mengubah Password*

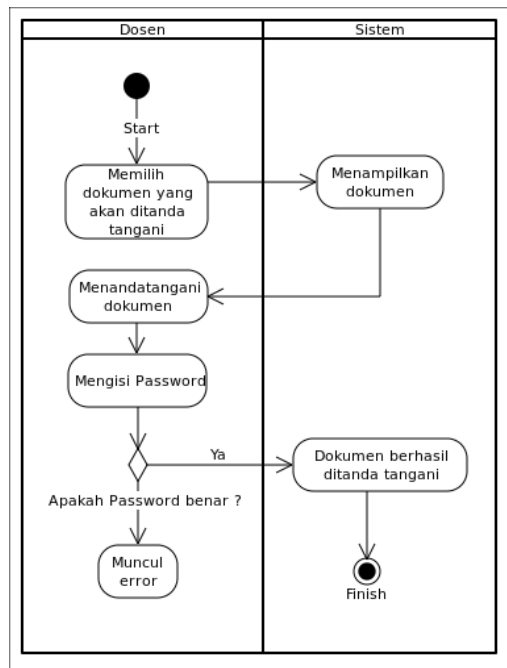
h. *Activity Diagram Melihat dokumen yang telah ditanda tangani*



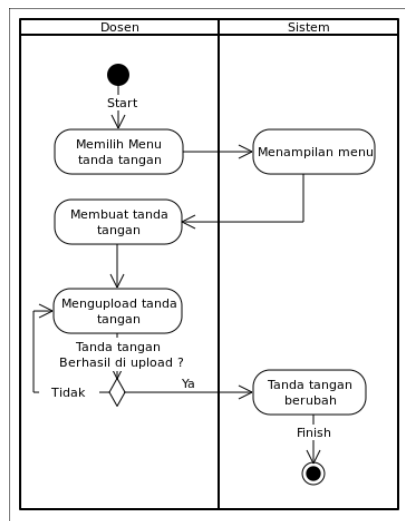
i. *Activity Diagram mendapat permintaan tanda tangan*



j. *Activity Diagram Menandatangani dokumen*

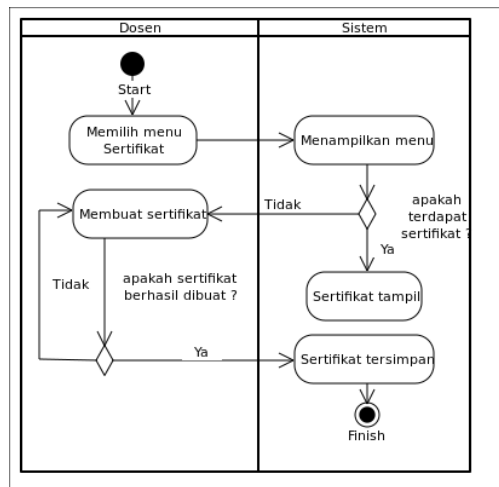


k. *Activity Diagram Membuat tanda tangan*



1. *Activity Diagram Membuat Sertifikat*





- m. *Activity Diagram Mengupload file*
- n. *Activity Diagram Melihat Progres*
- o. *Activity Diagram Mendownload Dokumen*

4.3.3.

#### 4.3.4. Simulasi Perhitungan Manual Algoritma RSA

Pada tahap ini penulis akan menjelaskan bagaimana perhitungan manual algoritma RSA yang sebelumnya telah dijelaskan pengertainya pada BAB 2. Serta akan memberikan contoh kasus dalam kehidupan nyata.

Sebagai contoh, jika seorang *user A* ingin mengirim pesan ke *user B*, maka pertama kali yang harus dilakukan *user A* adalah mengirimkan *public key* miliknya kepada *user B*. setelahnya *user B* akan melakukan enkripsi pesan yang akan dikirimkan dengan menggunakan *public key user B*. lalu *user B* melakukan enkripsi dengan cara  $C = M^e \pmod n$ ,  $C$  adalah *ciphertext* yang dikirimkan dan  $M$  adalah *message* atau pesan. Setelah itu *user A* akan menerima pesan  $C$  dan melakukan dekripsi dengan cara  $M = C^d \pmod n$ . perlu diperhatikan bahwa panjang pesan  $M$  tidak harus lebih kecil dari  $n$ .

Contoh perhitungan dan tahap-tahap yang akan dilakukan

1. Tahap ekspansi kunci
  - Pilih dua buah bilangan prima yang berlainan  
 $p = 61$  dan  $q = 53$

- Hitung  $n = pq$   
 $n = 61 * 53 = 3233$
- Hitung *totient*  $\phi(n) = (p-1)(q-1)$   
 $\phi(n) = (61-1)(53-1) = 3120$
- Tentukan  $e > 1$  yang *coprime* dengan 3120  
 $e = 17$
- Pilih  $d$  sehingga memenuhi  $ed = 1 \bmod \phi(n)$   
 $d = 2753$   
 $17 * 2753 = 46801 = 1 + 15 * 3120$

## 2. Tahap enkripsi

- Jika  $m = 123$ , maka proses enkripsi adalah :  
 $c = 123^{17} \bmod 3233 = 855$

## 3. Tahap dekripsi

- Jika  $c = 855$ , proses dekripsi adalah:  
 $m = 855^{2753} \bmod 3233 = 123$

Contoh lain misalkan *pesan (plaintext)* yang akan dikirimkan adalah :  $m = \text{"HARI INI"}$  atau dalam sistem decimal (pengkodean ASCII) adalah 7265827332737873

Cara penyandiannya adalah :

1. Pecah  $m$  menjadi blok yang lebih kecil, misalnya  $m$  dipecah menjadi 6 blok yang berukuran 3 digit :

$$m_1 = 726 \qquad m_4 = 273$$

$$m_2 = 582 \qquad m_5 = 787$$

$$m_3 = 733 \qquad m_6 = 003$$

Nilai-nilai  $m$  ini masih terletak di dalam selang  $[0, 3337 - 1]$  agar transformasi menjadi satu kesatuan.

2. Jika kunci public adalah  $e = 79$  dan  $n = 3337$ , maka blok-blok *plaintext* dapat dienkripsi menjadi:

$$c_1 = 726^{79} \bmod 3337 = 215; \qquad c_4 = 273^{79} \bmod 3337 = 776;$$

$$c_2 = 582^{79} \bmod 3337 = 1743; \qquad c_5 = 787^{79} \bmod 3337 = 933;$$

$$c_3 = 733^{79} \bmod 3337 = 1731; \qquad c_6 = 003^{79} \bmod 3337 = 158$$

jadi *ciphertext* yang dihasilkan adalah :

$$c = 215\ 776\ 1743\ 933\ 1731\ 158$$

3. Deskripsi pesan, dilakukan dengan menggunakan kunci pribadi  $d = 1019$

Blok-blok *ciphertext* dideskripsikan menjadi:

$$m_1 = 215^{1019} \bmod 3337 = 726$$

$$m_2 = 776^{1019} \bmod 3337 = 582$$

$$m_3 = 1743^{1019} \bmod 3337 = 733$$

$$m_4 = 933^{1019} \bmod 3337 = 273$$

$$m_5 = 1731^{1019} \bmod 3337 = 787$$

$$m_6 = 158^{1019} \bmod 3337 = 003$$

akhirnya diperoleh Kembali *plaintext* semula

$m = 7265827332737873$ , yang dalam sistem pengkodean ASCII adalah

$$m = \text{HARI INI}$$

#### 4.4. Implementasi Sistem

Dalam tahap implementasi sistem ini penulis akan menjelaskan tahapan demi tahapan dalam metode pengembangan *software Scrum* yaitu, sebelum membuat *product backlog* dilakukan analisis fitur yang akan dibuat pada pengembangan aplikasi. Selanjutnya dari analisis tersebut dibuatkan *product backlog*. *Product Backlog* merupakan daftar fitur yang akan dibuatkan dalam mengembangkan sistem. Selanjutnya setelah *product backlog* dibuat kemudian masuk ke dalam tahapan *sprint*. Dalam tahap *sprint* terdiri dari beberapa *event* yaitu, *Sprint Planning*, *Daily Scrum*, *Sprint Review*, dan *Sprint Retrospective*.

Penjabaran dari tahap-tahap implementasi metodologi *Scrum* pada pengembangan sebagai berikut:

##### 4.4.1. Analisis Fitur pada Pengembangan Sistem

Pada tahap ini *product owner* menceritakan secara detail semua fungsi yang nantinya ingin diterapkan pada sistem. Semua cerita dari

*product owner* penulis sebut sebagai *User stories* dan selanjutnya dilakukan pendataan secara menyeluruh beserta dengan prioritas masing-masing *user stories* untuk menentukan seberapa pentingnya fungsi tersebut untuk diimplementasikan. Berikut *User stories* yang didapatkan:

No	Role	Saya ingin	Maka	Prioritas
1.	Semua	User dapat login menggunakan NIDN/NIM dan password.	Dibuatkan halaman login untuk pengguna dengan NIDN/NIM dan password.	<i>Hight</i>
2.	Semua	User dapat melakukan pendaftaran akun dengan email dari Universitas Pelita Bangsa.	Diutakan halaman untuk melakukan pendaftaran akun baru.	<i>Hight</i>
3.		Halaman utama ketika webiste dibuat.	Dibuatkan halaman <i>landing page</i> untuk halaman awal website	<i>Low</i>
4.	Semua	Dokumen yang telah di tanda tangani dapat diverifikasi keabsahannya.	Dibuatkan fitur untuk memverifikasi keabsahan tanda tangan pada dokumen skripsi.	<i>Medium</i>
5.	Semua	Terdapat halaman <i>dashboard</i> pada sistem.	Dibuatkan halaman <i>dashboard</i> pada sistem.	<i>Low</i>

6.	Semua	User dapat mengubah semua informasi profilnya sendiri	Dibuatkan halaman profile untuk melihat informasi tiap user dan untuk mengubah informasi yang ada.	<i>Low</i>
7.	Semua	User dapat mengubah password yang digunakan untuk login ke sistem.	Dibuatkan halaman untuk merubah password.	<i>Medium</i>
8.	Semua	User dapat melihat dokumen yang telah diupload.	Dibuatkan fitur untuk melihat dokumen yang telah diupload.	<i>Medium</i>
9.	Dosen	Dapat menandatangani dokumen.	Dibuatkan halaman untuk meninjau dokumen kemudian menandatangani dokumen tersebut.	<i>Hight</i>
10.	Dosen	Dapat membuat tanda tangan.	Dibuatkan halaman untuk membuat tanda tangan.	<i>Medium</i>
11.	Dosen	Dapat membuat sertifikat sebagai identitas penandatangan dokumen.	Dibuatkan halaman untuk membuat sertifikat.	<i>Hight</i>
12.	Dosen	Dapat menerima notifikasi	Dibuatkan fitur notifikasi untuk	<i>Medium</i>

		permintaan tanda tangan.	permintaan tanda tangan yang masuk.	
13.	Mahasiswa	Dapat mengupload dokumen skripsi berupa file pdf.	Dibuatkan halaman upload dokumen skripsi.	<i>Hight</i>
14.	Mahasiswa	Dapat melihat progres penandatanganan dokumen skripsi.	Dibuatkan fitur untuk mengetahui progres penandatanganan dokumen.	<i>Low</i>
15.	Mahasiswa	Dapat mengunduh/mendownload dokumen skripsi.	Disediakan fitur untuk mengunduh dokumen skripsi.	<i>Medium</i>

#### 4.4.2. Product Backlog

Pada tahap ini setelah didapatkan *user stories* dari *product* selanjutnya adalah pembagian *product backlog* serta tahapan *sprint* dari *user stories* tersebut. Prioritas suatu *user stories* menentukan posisi *sprint*. Dalam pengembangan sistem dibagi menjadi 4 *sprint*, berikut adalah tabel *Sprint* pada proses pembuatan sistem ini:

Task Name	Prioritas	Estimasi (hari)	Aceptance Criteria
<b>Sprint 1</b>			
Login Page	<i>Hight</i>	1	<ul style="list-style-type: none"> <li>Halaman <i>Login</i> muncul ketika menu <i>Login</i> dipilih</li> <li>Dapat memasukkan data yang diminta</li> </ul>

			<ul style="list-style-type: none"> <li>• Ketika menekan <i>Login</i> maka akan melakukan <i>validasi</i> data</li> <li>• Ketika data sesuai maka <i>Login</i> berhasil</li> </ul>
Register Page	<i>Hight</i>	2	<ul style="list-style-type: none"> <li>• Halaman Daftar ditampilkan ketika memilih menu Daftar di halaman <i>Login</i></li> <li>• Dapat memasukkan data yang diminta</li> <li>• Ketika menekan <i>Sign-Up</i> maka akan melakukan <i>validasi</i> data</li> <li>• Ketika sudah melakukan Daftar maka akan mendapat <i>link</i> untuk aktivasi akun</li> </ul>
Upload Page	<i>Hight</i>	3	<ul style="list-style-type: none"> <li>• Halaman <i>Upload</i> muncul ketika menekan menu <i>Upload</i></li> <li>• <i>User</i> dapat mengisi data yang diminta</li> <li>• <i>User</i> dapat melakukan tahapan <i>upload</i> dokumen</li> <li>• Ketika tahapan selesai maka dokumen berhasil diupload</li> </ul>

			<ul style="list-style-type: none"> <li>• Muncul pesan pada layar ketika berhasil <i>upload</i> dokumen</li> </ul>
Uploaded Page	<i>Hight</i>	1	<ul style="list-style-type: none"> <li>• Halaman dokumen di <i>upload</i> muncul ketika menu skripsi diupload dipilih</li> <li>• Muncul daftar dokumen ketika <i>user</i> pernah melakukan <i>upload</i> dokumen</li> </ul>
Signing Page	<i>Hight</i>	1	<ul style="list-style-type: none"> <li>• Halaman penandatanganan dokumen muncul ketika tombol Tanda Tangani ditekan beserta detail dokumen yang akan ditanda tangani</li> <li>• <i>User</i> dapat melakukan penandatanganan dengan mekenan tombol tanda tangani</li> <li>• <i>User</i> harus mengisi password untuk menandatangani dokumen</li> <li>• Ketika dokumen berhasil ditanda tangani akan muncul pesan berhasil</li> </ul>



			<ul style="list-style-type: none"> <li>• Muncul pesan eror ketika terdapat kesalahan ketika menanda tangani dokumen</li> </ul>
<b>Sprint 2</b>			
Certificate Page	<i>Hight</i>	3	<p>Halaman pembuatan sertifikat muncul ketika menu sertifikat dipilih</p> <p><i>User</i> dapat membuat sertifikat dengan menekan tombol buat sertifikat</p> <p>Muncul pesan ketika berhasil membuat sertifikat</p> <p>Muncul pesan eror ketika terjadi kesalahan</p>
Fitur Verification	<i>Medium</i>	2	<p><i>User</i> dapat mengupload dokumen untuk di verifikasi</p> <p><i>User</i> dapat melakukan verifikasi tanda tangan dengan menekan tombol verifikasi</p> <p>Muncul hasil verifikasi ketika dokumen selesai divetifikasi.</p>
Create Sign Page	<i>Medium</i>	2	Halaman pembuatan tanda tangan muncul ketika

			<p>menu Tanda Tangan ditekan</p> <p><i>User</i> dapat membuat tanda tangan</p> <p><i>User</i> dapat mengunduh tanda tangan sesuai dengan format yang dipilih</p> <p><i>User</i> dapat mengupload tanda tangan yang baru dibuat</p> <p>Muncul pesan ketika tanda tangan berhasil diunggah</p> <p>Muncul pesan <i>error</i> ketika terjadi kesalahan</p>
Notification	<i>Medium</i>	1	Muncul notifikasi ketika datang permintaan tanda tangan
Sprint 3	<i>Medium</i>		
Change Password page	<i>Medium</i>	1	<p>Muncul halaman profile ketika tombol profile ditekan</p> <p><i>User</i> dapat mengisi data setelah tombol ubah <i>password</i> ditekan</p> <p>Muncul pesan ketika password berhasil dibuat</p> <p>Muncul pesan <i>error</i> ketika terjadi kesalahan</p>
Download	<i>Medium</i>	1	Detail dokumen muncul ketika dokumen dipilih

			<i>User</i> dapat mengunduh dokumen yang telah dipilih
Documen view	<i>Medium</i>	1	Muncul daftar dokumen yang ada ketika menu dokumen dipilih <i>User</i> dapat menekan tombol <i>detail</i> untuk melihat dokumen yang telah dipilih
Change Profile page	<i>Low</i>	1	Halaman profile muncul ketika menu profile dipilih <i>User</i> dapat mengubah data profile Muncul pesan ketika berhasil merubah data Muncul pesan error ketika terjadi kesalahan
Sprint 4			
Dashboard Page	<i>Low</i>	2	Muncul halaman dashboard ketika menu Home dipilih
Progress view	<i>Low</i>	2	Terdapat tanda tangan pada setiap dokumen yang diunggah
Landing Page	<i>Low</i>	2	Muncul halaman utama <i>website</i> ketika pertama kali mengunjungi <i>webiste</i>

#### 4.4.3. Sprint

Pada tahap ini penulis akan memulai proses selanjutnya yaitu adalah *sprint*. Sesuai dengan tabel sprint yang telah dibuat sebelumnya, masing-masing *sprint* terdiri dari 3 (tiga) sampai 5 (lima) *product backlog*. Tahapan-tahapan sprint akan penulis jabarkan sebagai berikut:

#### **1. Sprint 1 (satu)**

Pada *Sprint* 1 (satu) ini penulis akan menjelaskan daftar pekerjaan berdasarkan *product backlog* yang telah dibuat. Berikut adalah penjelasan *scrum event* pada *sprint* 1 (satu);

- a. Task 1 Pembuatan .....
- b. Task 2 pembuatan .....

## Daftar Pustaka

- [1] Kurniawan, Yusuf, 2004, Kriptografi: Keamanan Internet dan Jaringan Komunikasi, Bandung: Informatika
- [2] Munir, Rinaldi. 2006. Kriptografi. Program Studi Teknik Informatika Institut Teknologi Bandung
- [3] Dwiperdana, Aditia, "Cryptographic Hash Function dan Penggunaannya Dalam Digital Signature", Institut Teknologi Bandung. 2007. Tersedia dalam pada <http://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2006-2007/Makalah/Makalah0607-42.pdf>
- [4] Ireland, David, 2010, *RSA Algoritma*, [https://www.dimgt.com.au/rsa\\_alg.html#simpleexample](https://www.dimgt.com.au/rsa_alg.html#simpleexample). Diakses 04 Juli 2021
- [5] FIPS 180-3, 2008, Secure Hash Standard (SHS), USA: Information Technology Laboratory, National Institute of Standard and Technology
- [6] Stallings, William, 1995, *Network and Internetwork Security Principles and Practice*, New Jersey: Prentice Hall Inc.
- [7] <https://pypi.org/project/cryptography/>
- [8] Rahadian Irsyad, "Penggunaan Python Web Framework Flask Untuk Pemula", Laboratorium Telematika, Sekolah Teknik Elektro & Informatika, Institut Teknologi Bandung
- [9] [Nisa Hanum Harani](#), [Miftahul Hasanah](#), "Deteksi Objek dan Pengenalan Karakter Plat Nomor Kendaraan Indonesia Berbasis Python", Bandung. Kreatif Industri Nusantara
- [10] van Rossum, Guido (1993). "An Introduction to Python for UNIX/C Programmers". Proceedings of the NLUUG najaarsconferentie (Dutch UNIX users group). even though the design of C is far from ideal, its influence on Python is considerable.
- [11] <https://pypi.org/project/pyHanko/>
- [12] Hidayatullah, Priyanto, dan Jauhari Khairul Kawistara. 2017. Pemrograman WEB. Bandung. Informatika Bandung.

- [13] Sibero, Alexander F.K . 2013. Web Programming Power Pack. Yogyakarta : Mediakom.
- [14] (Sopian Handianto, *Membangun Responsive Webiste dengan Twitter Bootstrap 2.0+PHP dan MySQL*, 2014 hal:8)
- [15] Widyantoro. Wahyu.Buku Panduan Bootstrap. Margotekno
- [16] Chonoles, Michael Jesse, James A, Schardt (2003),UML 2 for Dummies, Wiley Publishing, New York
- [17] Henderi. 2008. Unified Modelling Language. Tangerang : Raharja Enrichment Centre (REC).
- [18] Rosari, R. W. 2008. PHP dan MySQL untuk pemula, Yogyakarta: Penerbit ANDI
- [19] [https://id.wikipedia.org/wiki/Portable\\_Document\\_Format](https://id.wikipedia.org/wiki/Portable_Document_Format)
- [20] W. Stalling, *Cryptography and Network Security Principles and Practices*. Fourth Edition, New Jersey: Pearson Education, Inc. 2005.
- [21] Munir, Rinaldi, *Public Key Infrastructure*, Institut Teknologi Bandung, 2004