Doc vs Library

94.26% Originality	5.74% Similarity	49 Sources

Library sources: 49 sources found

1411510918_WisnuAbdulHalim(1).doc	1.25%
1411510645-YoshuaAndrewHoutama-Gasal-2019-HariSoetanto.docx	1.25%
1211511256_MuhammadGunawanRasyid_Genap_2018 (1) (1).doc	0.91%
Jurnal Tesis - Indonesia.pdf	0.83%
1311511107_fazarsidik_Gasal_2018 (1).doc	0.77%
1311500753_Iskandar zulkarnain.doc	0.77%
1411503772_DwiSeptian_Genap_2017_PipinFaridaAriyani.docx	0.72%
1411504101_Mohamad Rizal_Genap_2018_Painem (1).docx	0.67%
1411504101_Mohamad Rizal_Genap_2018_Revisi.docx	0.67%
1411504101_Mohamad Rizal_Genap_2018_Painem.docx	0.67%
1411504101_Mohamad Rizal_Genap_2018.docx	0.67%
1411504101_Mohamad Rizal_Genap_2018_Revisi.docx	0.67%
1511502716-EdoFirnanda-Genap-2018.docx	0.61%
1412500868-AlifFaturchmanFauzi-Genap-2017-Samsinar.doc	0.59%
1412500694-FanisyaOktarina-Genap-2018-AdyWidjaja.docx	0.59%
1512530237-LirisMardiyanti-Genap-2018-AdyWidjaja.doc	0.59%
1312530049-AhmadSiddiq-Genap-2018-AdyWidjaja.doc	0.59%
1411500166_Fitriyanto_Genap_2018.pdf	0.59%
1312502923_Rivaldi Nur Hudha_Genap_2018.docx	0.59%
1412500413_Abdilah Sanad_Genap_20172018_Lusi Fajarita.docx	0.59%
1412500744_SunnyChristinCahyadi_Genap_20172018_HumisarHasugi.docx	0.59%
1311500357-SriMutiaraInsani-GENAP-2017.docx	0.59%
1412503094-NitaNatasya-Genap-2018-AdyWidjaja.docx	0.59%
1411530098_AdeIrmawan_Genap_2017.doc	0.59%
1412530204-MartinusAlberto-Gasal-2018.doc	0.59%
1412530311-AidaPrastiwiPutri-Gasal-2018.doc	0.59%
1412504514_Aldi Saputra.docx	0.59%
1312530353-Husain-Gasal-2018-AdyWidjaja.doc	0.59%
1411500802-ibnu-hardiyatna (1).doc	0.59%
1411503566_MochNovryanFirdaus_Genap_2018.docx	0.59%
Jurnal ilmiah Prapemrosesan jom_Teddy_21Agustus2018-rev1.docx	0.59%
1411501131_FakrianRahmanuli_2017_v2.doc	0.59%
19-2016-Prototipe Aplikasi Layanan Pengaduan Masyarakat.pdf	0.53%
1412500835-MuhammadAdibFuadi-Genap-2017.docx	0.51%

Similarity
Similarity from a chosen source
abc Possible character replacement
Citation
References

jurnal_1312510561_prayogi.docx	0.51%
JURNAL 1411500505 HIDAYATUSSALAM.docx	0.51%
1412501288-ZulhamdiApriyadi-Genap-2017-Samsinar.doc	0.51%
14115001891_ArielBudiman_Genap_20172018.docx	0.51%
1411503525_Akhifadly_Genap_2018.docx	0.51%
1411500992_Febriansyah Eko Saputro_Genap20172018.pdf.pdf	0.45%
1211520109_AnggaLendrasidi_Genap_2018.docx	0.37%
1411600679_Fikri Al Farasyi.docx	0.35%
1211510829_FirmanMujahidin_Gasal_2017_Subandi.docx	0.29%
1411502881_AchmadFachriJohaniAnanto_Genap_2018(1).docx	0.29%
1411502006-Firas Umar Tsabit-Genap-2017-Purwanto.docx	0.27%
1412504118-PeniTriRahayu-Genap-20172018-Yudi Santoso.pdf	0.27%
1411530148_FIRDIANSYAH_Gasal_2018_Siswanto.doc	0.27%
3. Jurnal Tesis - Indonesia (3).pdf	0.24%
1311502163-ArisKrisnaMarantika-Genap-20172018-SejatiWaluyo.doc	0.21%

IMPLEMENTASI KRIPTOGRAFI ALGORITMA *RSA* UNTUK MENGAMANKAN DATA PADA SISI *WEB SERVICE* APLIKASI BLANJA QUICKPAY

Ferat Muzadid[1)], Hari Soetanto[)]
[1]Program studi, Fakultas Teknologi Informasi, Universitas Budi Luhur
[1,2]Jl. Raya Ciledug, Petukangan Utara, Kebayoran Lama, Jakarta Selatan 12260
E-mail : feratmuzadid@gmail.com[1)], hari.soetanto@budiluhur.ac.id[2)]

Abstrak

Aplikasi blanja quikpay adalah selah satu program monile yang bergerak pada bidang retail online,pada aplikasi ini terdapat data – data yang bersifat penting terutama data customer yang sudah menjadi user aplikasi blanja quikpay. Pada aplikasi tersebut belum dilengkapi dengan pengamanan data sehingga data – data penting tersebut akan sangat mudah dicuri maupun diubah oleh pihak lain yang tidak bertanggung jawab. Oleh karena itu pentingnya sebuah metode pengamanan data pada aplikasi tersebut supaya data – data penting tersebut dapat terjaga keamanannya. Untuk mengamankan data – data tersebut perlu di terapkan sebuah metode pengamnan data yang sering disebut dengan kriptografi, ada dua hal yang sangat penting pada kriptografi yaitu proses enkripsi dan proses dekripsi, pada proses enkripsi dan dekripsi dibutuhkan suatu algoritma untuk melakukan proses enkripsi dan dekripsi, berdasarkan kasus diatas algoritma yang sesuai untuk memecahkan permasalahan yang ada yaitu algoritma RSA, karena algoritma RSA memiiliki dua pembangkit kunci (kunci public dan kunci private) sesuai dengan aplikasinya yang berbasis client server sehingga membutuhkan dua kunci yang berbeda. Penerapan kriptografi pada aplikasi ini dilakukan pada sebuah web service, karena pada web service tersebut sangat riskan terhadap pencurian data. Adapun tujuan di terapkannya metode kriptografi dengan algoritma RSA pada web service ini yaitu untuk mengamankan data yang bersifat penting, sehingga nantinya para customer akan nyaman menggunakan aplikasi blanja quikpay ini tanpa ada rasa khawatir dengan data – data yang disimpan pada aplikasi ini.

Kata kunci: Web Service, RSA, Kriptografi, Algoritma.

1. Pendahuluan

1.1. Latar Belakang Masalah

Keamanan data merupakan hal yang sangat penting dalam menjaga kerahasiaan informasi terutama yang berisi informasi sensitif yang hanya boleh diketahui isinya oleh pihak yang berhak saja, apalagi jika pengirimannya dilakukan melalui jaringan publik, apabila data tersebut tidak diamankan terlebih dahulu, akan sangat mudah disadap dan diketahui isi informasinya oleh pihak-pihak yang tidak berhak. Salah satu cara yang

digunakan untuk pengamanan data adalah menggunakan sistem kriptografi yaitu dengan menyandikan isi informasi (plaintext) tersebut menjadi isi yang tidak dipahami melalui proses enkripsi dan untuk memperoleh kembali informasi yang asli, dilakukan proses dekripsi, disertai dengan menggunakan kunci yang benar. Namun, sejalan dengan perkembangan ilmu penyandian atau kriptografi, usaha-usaha untuk memperoleh kunci tersebut dapat dilakukan oleh siapa saja, termasuk pihak yang tidak sah untuk memiliki informasi tersebut. Oleh karena itu, penelitian tentang

kriptografi akan selalu berkembang untuk memperoleh algoritma kriptografi yang makin kuat, sehingga usaha-usaha untuk memecah kode kriptografi secara tidak sah menjadi lebih sulit.

1.2. Perumusan Masalah

Rumusan masalah dalam penelitian ini adalah sebagai berikut :

a. Ditemukannya celah untuk *internal or external attacker* melakukan pencurian data transaksi, pola pesan *transaksi, user and password* pada aplikasi.
b. Bagaimana mengamankan transaksi pembelian dan pembayaran dari pencurian informasi dengan cara mengimplementasikan kriptografi dengan menggunakan metode *RSA* (Enkripsi).
c. Bagaimana cara mendekripsi transaksi pembayaran yang sudah terenkripsi menjadi pesan yang dapat di kenali oleh *server* tanpa mengalami perubahan isi sedikitpun.

1.3 Tujuan Dan Manfaat

Tujuan dari penelitian ini adalah sebagai berikut:

a. Menjelaskan teknik menyisipkan pesan teks pada gambar.
b. Menerapkan metode *RSA* untuk mengenkripsi dan dekripsi pada web service.

Manfaat dari penelitian ini adalah agar nantinya pengguna dapat menggunakan aplikasi ini dengan nyaman tanpa khawatir pencurian data.

2 TEORI DAN METODE PENELITIAN

2.1. Kriptografi

Cryptographic system atau *cryptosystem* adalah suatu fasilitas untuk mengkonversikan *plaintext* ke *ciphertext* dan sebaliknya. Dalam sistem ini, seperangkat parameter yang menentukan *transformed* pencipheran tertentu disebut suatu set kunci. Proses enkripsi dan dekripsi diatur oleh satu atau beberapa kunci kriptografi. Secara umum, kunci-kunci yang digunakan untuk proses pengenkripsian dan pendekripsian tidak perlu identik, tergantung pada sistem yang digunakan.

Secara umum operasi enkripsi dan dekripsi dapat diterangkan secara matematis sebagai berikut:

EK(M)=C (Proses Enkripsi) (1)

DK(C)=M (Proses Dekripsi)

Pada saat proses enkripsi kita menyandikan pesan M dengan suatu kunci K lalu dihasilkan pesan C. Sedangkan pada proses dekripsi, pesan C tersebut diuraikan dengan menggunakan kunci K sehingga dihasilkan pesan M yang sama seperti pesan sebelumnya.

2.2. Algoritma RSA

RSA merupakan algoritma kriptografi asimetri, dimana kunci yang digunakan untuk mengenkripsi berbeda dengan yang digunakan untuk mendekripsi. Kunci yang digunakan untuk mengenkripsi disebut dengan kunci *public*, dan yang digunakan untuk mendekripsi disebut dengan kunci privat. *RSA* adalah salah satu algoritma kriptografi yang menggunakan konsep kriptografi kunci publik. *RSA* membutuhkan tiga langkah dalam prosesnya, yaitu pembangkitan kunci, enkripsi, dan dekripsi. Proses enkripsi dan dekripsi merupakan proses yang hampir sama. Jika bilangan acak yang dibangkitkan kuat, maka akan lebih sulit untuk melakukan *cracking* terhadap pesan.

2.5 Perumusan Algoritma *RSA*

Algoritma *RSA* didasarkan pada teorema Euler (lihat bahan kuliah Teori Bilangan) yang menyatakan bahwa :

a) harus relatif prima terhadap n

b) Ø(n) = n(1 - 1/p1)(1 – 1/p2) ... (1 - 1/pr), yang dalam hal ini p1, p2, pr adalah faktor prima dari n.

c) Ø(n) adalah fungsi yang menentukan berapa banyak dari bilangan-bilangan 1, 2, 3, n yang relatif prima terhadap n.

d) Berdasarkan sifat a k 0 b k (mod n) untuk k bilangan bulat 3 1, maka persamaan dapat ditulis menjadi

$$a^{K\emptyset(n)} = 1k \pmod n \quad (2)$$

atau

$$a^{K\emptyset(n)} = 1 \pmod n \quad (3)$$

e) Bila a diganti dengan m, maka persamaan menjadi

$$m^{K\emptyset(n)} = 1 \pmod n \quad (4)$$

f) Berdasarkan sifat ac o bc (mod n), maka bila persamaan dikali dengan m menjadi:

$$m^{K\emptyset(n)+1} = m \pmod n \quad (5)$$

yang dalam hal ini m relatif prima terhadap n.

g) Misalkan e dan d dipilih sedemikian sehingga

$$e \times d = 1 \pmod{\emptyset(n)} \quad (6)$$

atau

$$e \times d = k\emptyset(n) + 1 \quad (7)$$

h) Sulihkan (7) ke dalam persamaan (5) menjadi:

$$m^{e \times d} = m \pmod n \quad (8)$$

i) Persamaan (2.7) dapat ditulis kembali menjadi

$$(m^e)^d = m \pmod n \quad (9)$$

yang artinya, perpangkatan m dengan e diikuti dengan perpangkatan dengan d menghasilkan kembali m semula.

j) Berdasarkan persamaan (2.8), maka enkripsi dan dekripsi dirumuskan sebagai berikut:

$$E_e(m) = c = m^e \bmod n \quad (10)$$

$$D_d(c) = m = c^d \bmod n \quad (11)$$

k) Karena e x d = d x e, maka enkripsi diikuti dengan dekripsi ekivalen dengan dekripsi diikuti enkripsi :

$$D_d(E_e(m)) = E_e(D_d(m)) = m^d \bmod n \quad (12)$$

Oleh karena $m^d \bmod n = (m + jn)^d \bmod n$ untuk sembarangan bilangan bulat j, maka tiap plainteks m, m + n, m + 2n, ..., menghasilkan *ciphertext* yang sama. Dengan kata lain, transformasinya dari banyak ke satu. Agar tansformasinya satu-ke-satu, maka m harus dibatasi dalam himpunan {0, 1, 2, ..., n -1} sehingga enkripsi dan dekripsi tetap benar seperti pada persamaan (9) dan (10).

2.4 Keamanan RSA

Keaman algoritma *RSA* didasarkan pada sulitnya memfaktorkan bilangan besar menjadi factor-faktor primanya.
Masalah pemgfaktoran: Faktorkan n, yang dalam hal ini n adalah hasil kali dari dua atau lebih bilangan prima. Pada *RSA*, masalah pemfaktoran berbunyi: Faktorkan n menjadi dua factor primanya, p dan q, sedemikian sehingga n = p x q.

Sekali n berhasil difaktorkan menjadi p dan q, maka Ø(n) =(p - 1) dapat dihitung. Selanjutnya, karena kunci enkripsi e diumumkan (tidak rahasia), maka kunci dekripsi d dapat dihitung dari persamaan e x d ≡1 (mod Ø(n)). Penemu algoritma RSA menyarankan nilai p dan q panjang lebih dari 100 digit. Dengan demikian hasil kali n = p x q akan berukuran lebih dari 200 digit.

Menurut Riverst dan kawan-kawan, usah untuk mencari factor prima dari bilangan 200 digit membutuhkan waktu komputasi selama 4 milyar tahun, sedangkan untuk bilangan 500 digit membutuhkan waktu 10^{25} Tahun! (dengan asumsi bahwa algoritma yang tercapat saat ini dan komputer yang dipakai mempunyai kecepatan 1 milidetik).

Untunglah algoritma yang paling mangkus untuk memfaktorkan bilangan yang besar belum ditemukan. Selama 300 tahun para matematikawan mencoba mencari factor bilangan yang besar namun tidak banyak membuahkan hasil. Semua bukti yang diketahui menunjukan bahwa upaya pemfaktoran itu luar biasa sulit.

Factor ini lah yang membuat algoritma *RSA* tetap dipakai hingga saat ini. Selagi belum ditemukan algortma yang mangkus untuk memfaktorkan bilangan bulat menjadi factor primanya, maka algoritma *RSA* tetap direkomendasikan untuk mengenkripsi pesan.

2.6 Web Service

a. Pengertian Web Service

Web Service adalah suatu sistam prangkat lunak yang dirancang untuk mendukung interoperabilitas dan interaksi antara sistem pada suatu jaringan. *Web Service* diartikan sebagai sebuah antar muka (interface) yang menggambarkan sekumpulan operasi yang dapat diakses melalui jaringan, misalnya internet dalam bentuk pesan XML (*eXtensible Markup Language*). *Web Service* adalah aplikasi perangkat lunak yang tersedia pada web yang melaksanakan fungsi yang spesifik (Wulandari dan Wicaksana 2006). Sedangkan menurut Michael C. Daconta (2005), *Web Service* adalah aplikasi perangkat lunak yang dapat ditemukan, diuraikan dan diakses berdasarkan pada XML (*eXtensible Markup Language*) dan protocol standard web pada internet, extranet dan internet.

Web Service menyediakan *standard* komunikasi diantara berbagai aplikasi *software* yang berbeda-beda, dan dapat berjalan berbagai *platform* maupun *framework*. *Web Service* digunakan sebagai suatu fasilitas yang disediakan oleh suatu web untuk menyediakan layanan (dalam bentuk informasi) kepada sistem lain, sehingga sistem lain dapat berinteraksi dengan sistem tersebut melalui layanan-layanan (*service*) yang disediakan oleh suatu sistem yang menyediakan *Web service.* Teknologi pada *web service* dapat mengubah kemampuan *transactional web,* yaitu kemapuan *web* untuk saling berkomunikasi dengan pola *program – to - program* (*P2P*). Fokus *web* selama ini di dominasi oleh komunikasi *program – to – usher* dengan interaksi *business – to consumer* (*B2C*), sedangkan *transactional web* akan di dominasi oleh *program – to – business* (Ghifari dan karya, 2011).

Web service sebenarnya adalah kumpulan dari fungsi dan metode yang terdapat pada sebuah *server* yang dapat dipanggil oleh *client* dari jarak jauh, kemudian untuk memanggil metode-metode tersebut kita bebas menggunakan aplikasi yang akan dibuat dengan Bahasa pemograman yang dijalankan pada *platform* apa saja (Marthasari, 2010). Adanya teknologi web *service* dapat menyembatani perbedaan-perbedaan teknologi dari masing-masing sumber dapat ditarik kesimpulan bahwa *web service* merupakan kumpulan layanan yang disediakan melalui jaringan berbasis *web* dengan *standard* yang telah ditetapkan mampu menunjang interoperabilitas dan dapat berjalan diberbagai *flatform* dan *framework*.

b. Metode Web Service

1) *Simple Object Access Protocol* (SOAP)

Menurut O'Reilly (2002) SOAP merupakan suatu protocol berbasis XML yang digunakan untuk kebutuhan pertukaran informasi dalam suatu sistem terdistribusi dan terdesentralisasi, seperti halnya *IIOP* (pada *COBRA*). *ORCP* (pada *DCOM*), dan *JRMP* (pada *RMI*). Berbeda dengan independent terhadap *platform,* model pemrograman, dan *Transport* SOAP dapat dikirimkan melalui HTTP, SMTP maupun FTP.

Pesan SOAP berbentuk sekumpulan XML *schema* yang mendefinisikan *format* untuk mentransmisikan pesan XML melalui jaringan, termasuk tipe data dan cara menstrukturkan pesan secara tepat sehingga dapat mudah dipahami oleh *server* atau *end-point* lainnya.

Pesan SOAP terdiri dari 3 bagian, yaitu :

a. *Envelope,* yaitu suatu kerangka yang mendefinisikan apa yang ada dalam pesan dan bagaimana pesan harus diproses serta menunjukan *recipient* dari *message* tersebut.

b. *Header,* yaitu berisi informasi yang berhungan dengan keamanan dan *rounting.* Keberadaan *header* dalam SOAP bersifat *optional.*
c. *Body,* yaitu berisi data yang berhubungan dengan aplikasi tertentu yang sedang dipertekarkan. Data di *mark-up* sebagai XML dan dimasukkan salam format yang spesifik yang didefinisikan dalam XML *Schema.*

2) *Web Service Description Language* (WSDL)

Menurut Simon St. Laurent (2002) WSDL merupakan bahasa *standard* yang digunakan mekanisme untuk mendeskripsikan *service* yang disediakan oleh sistem (*web service*) lokasi keberadaan *service* tersebut dan bagaimana cara memperolehnya secara struktur dalam *format* XML, WSDL dapat dianalogikan sebagai IDL (*Interface definition language*) dalam COBRA dan COM. *Service* dideskripsikan sebagai koleksi dari *entry-point* atau *port* komunikasi. WSDL mendeskripsikan *sevice* dengan menggunakan elemen sebagai berikut :

a) *Type,* yaitu tipe data yang digunakan sevahai argumen dan *return type.*
b) *Message,* yaitu merepresentasikan definisi data yang ditransmisikan.
c) *Port type,* yaitu sekumpulan oprasi yang didukung oleh satu atau lebih *end-port.*
d) *Binding,* yaitu mendifinisikan protocol dan *format* pertukaran data untuk operasi yang didefinisikan ole *port type.*
e) *Port,* yaitu mendefinisikan *end-point* yang digunakan untuk *binding.*
f) Service, yaitu menspesifikasikan *end-point* yang berkaitan dan sediakan oleh *web service.*
g) Operation, yaitu mendefinisikan kemampuan yang didukung oleh *service* tertentu.

3) Restful

Menurut CH Ram Mohan Reddy (2011) REST merupakan seingkatan dari *Representasi State Transfer. Rest service* adalah sebuah aplikasi SOA yang mempunyai karakteristik *stateless, client-server, cacheable communication protocol.* Protokol yang digunakan dalam komunikasi data *Rest* adalah menggunakan protokol HTTP, *RESTful web service* juga dikenal dengan nama *RESTful web* APUm merupakan sebuah *web service* yang di implementasikan dengan menggukan HTTP menggunakan *method* milik HTTP antara lain: *GET, PUT, POST, or DELETE.*

Bagian tersebut yang mendefinisikan bahwa *request* dari *client* ke *server* itu berupa *method*. Dengan begitu pihak *server* bias mendefinisikan perintah yang akan digunakan untuk merespon *GET* tersebut. Untuk *library simple RESTful* bias menggunakan *Simple-Rest.* Hanya di *custome* sesuai kebutuhan. Untuk *client,* bisa menggunakan salah satu ekstensi dari *google chrome* : *Simple-Rest-Client-Chrome.* Untuk penggunaan *method GET, POST, PUT,* atau *DELETE* bisa sesuai kebutuhan. Sebagai berikut :

a) *GET,* yaitu bisa digunakan apabila *request* ke *server* merupakan data yang sedikit. Karena *GET* ada keterbatasan data. Contoh penggunaan *URL* di *browser*
b) *POST,* yaitu biasa digunakan apabila pengiriman data dengan data yang banyak. Contoh data yang ada di *form* dengan *method post.*
c) *PUT,* yaitu bisa digunakan untuk pengiriman *file* melalui *service.* Contoh *upload file.*
d) *DELETE,* yaitu untuk yang satu ini sangat jarang digunakan, karena dapat diwakili oleh *GET*. Tapi, ada baiknya digunakan untuk proses penghapusan di data *server.* Untuk hasil *response* dari *server RESTful* dapat mengeluarkandata berupa Json, XML dan *text format*. Sehingga penggunaannya sesuai kesepakatan. Apakah akan menggunakan XML, JSON atau *text* sebagai pesan yang dikirimkan. *RESTful* memiliki kelemahan, yaitu ada standarisasi *format* pesan. Jadi untuk *format* pesan akan dibuat sesuai kebutuhan atau kesepakatan bersama. Berikut cara kerja *restful web service :*

Gambar 1 Konsep Web Service

1. ***Extreme Programming***

Nama *extreme programming* pertama kali diciptakan oleh Kent Beck (2002). Pendekatan ini dikembangkan dengan mendorong suatu kebiasaan yang baik seperti melakukan pengembangan secara terus menerus sampai mencapai tingkat *'extreme'*. Sebagai contohnya, dalam *extreme programming,* beberapa jenis sistem, dikemvangkan dengan banyak *programmer* yang terintegrasi dan dilakukan *test* dalam hari itu juga (Pressman 2010). Kunci aktivitas dari *extreme programming* disimpulkan menjadi 4 proses yaitu :

a. *Planning*

Tahap *planning* ini diawali dengan aktifitas 'mendengarkan'. Mendengarkan yang dimaksud disini ialah sebuah aktifitas mengumpulkan informasi mengenai kebutuhan dari aplikasi agar anggota-anggota teknikal dari tim *extreme programming* dapat memahami kontek bisnis dan juga untuk mendapatkan ketegasan dari *output* fitur utama dan fungsionalitasnya yang dihasilkan. Selanjutnya, tim *extreme programming* akan menentukan lamanya pengembangan dari tiap model di dalam aplikasi

.

b. *Design*

Design pada *extreme programming* mengikuti prinsip KIS (*Keep It Simple*). *Design* yang *simpe* selalu di utamakan ketimbang *design* yang lebih kompleks. Selain itu, tahap *design* menyediakan panduan untuk mengimplementasikan aplikasi sesuai perencanaan yang ada tanpa menambahkan atau mengurangi dari yang sudah ada (*nothing less, nothing more*). *Design* dengan fungsionalitas yang tinggi sangat dihindarkan (karena para *Depelover* mengamsusikan bahwa *design* tersebut akan dibutuhkan kemudian). Metode *extreme programming* menganjurkan untuk melakukan *refactoring,* yakni sebuah teknik konstruksi dan juga merupakan sebuah teknik untuk mengoptimisasi *design.* Fowler (2000) mendeskripsikan *refactoring* sebagai sebuah proses pergantian sistem dari *software* tersebut, dimana pergantian ini tidak mempengaruhi pihak eksternal yang tidak berkaitan dengan teknikal untuk meningkatkan struktur internal.

c. *Coding*

Kunci utama selama aktivitas *coding* adalah *pair programming. Extreme programming* merekomendasikan bahwa dua orang yang bekerja sama dalam sebuah computer untuk membuat penggalan *code.* Metode ini menyediakan sebuah mekanisme untuk menyelesaikan masalah secara *realtime* (dua orang lebih baik dari satu orang) dan menjamin kualitas juga secara *realtime* (*code* langsung di-*review* setelah dibuat).

Dalam praktiknya, setiap orang memiliki peran yang berbeda. Sebagai contohnya, oran yang satu akan memikirkan detail *coding* dari *design* yang dibuat sementara yang satu lagi memastikan *coding* yang dibuat mengikuti *standard* atau mungkin saja memastikan *coding* yang dibuat sesuai perencanaan awal.

d. *Testing*

Pada tahap ini dilakukan *testing* terhadap aplikasi yang dibuat pada tahap *coding.* Tujuan dari tahap ini ialah untuk memastikan semua fitur yang telah dirancang sesuai dan dapat digunakan dengan baik. Setelah itu, *customer test,* yang ditentukan oleh *customer* dan berfokus pada semua fitur dan fungsionalitas dalam sistem yang di-*review* juga oleh *customer*

2.6 Metode Pengembangan

a. Studi Literatur

Melalui studi ini penulis memperoleh data atau informasi dengan mengumpulkan, mempelajari dan membaca berbagai referensi baik itu dari buku-buku, jurnal, makalah, *internet* dan berbagai sumber lainya yang menunjang dalam penulisan ini.

b. Analisis Data

Menganalisa algoritma kriptografi *RSA*, serta teknik-teknik yang digunakan.

c. Perancangan Sistem

Merancang sistem program untuk mengimplementasikan algoritma *RSA*, dengan menggunakan bahasa pemrograman Java, HTML, CSS, JavaScript.

d. Pengujian Sistem

Melakukan pengujian terhadap program yang telah dirancang serta menyimpulkan hasil dari pengujian.

3. ANALISA DAN PERANCANGAN

3.1 Analisa Masalah

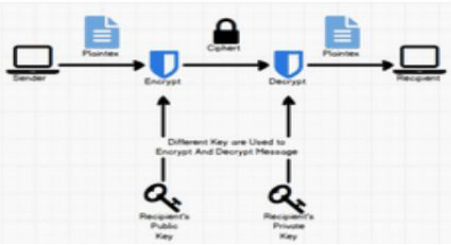
Perkembangan teknologi yang semakin pesat, mempermudah siapapun untuk melakukan transaksi pembelian dan pembayaran secara mudah dan cepat dalam bentuk digital. Penyediaan layanan pembelian dan pembayaran online produk digital menjadi hal yang sangat penting guna pempermudah pelayanan bagi siapapun dan dimanapun. Aplikasi dengan keamanan yang baik dalam bentuk *mobile* dan terintergrasi dengan server menggunakan *webservice* secara online sangat diperlukan untuk mempermudah akses bagi setiap konsumen yang berkendala dengan waktu dan tempat untuk sekedar membeli dan membayar produk digital secara langsung.

3.2 Pemecahan Masalah

Dari Analisa masalah, perlu adanya sebuah aplikasi berbasis android *mobile online* yang menghubungkan antara pengguna dengan berbagai macam informasi yang berkaitan dengan pembelian dan pembayaran produk digital, sehingga dapat mempermudah dalam pelayanan kepada masyarakat atau konsumen luas. Maka dibuatlah Aplikasi android *online* Belanja *Quickpay* agar konsumen dapat melakukan pembelian dan pembayaran produk digital melalui genggaman yang bisa dilakukan kapan saja dan dimana saja sesuai kebutuhan. Aplikasi Belanja *Quickpay* dibangun dengan metode enkripsi kriptografi RSA pada data yang berjalan di *web service* dan dengan antarmuka aplikasi yang menarik. Sehingga pengguna dapat merasa aman, nyaman dan mudah mengunakannya dalam kehidupan sehari-hari.

3.3 Arsitektur Kriptografi RSA

Untuk dapat memahami konsep kriptografi yang akan digunakan pada aplikasi dapat melihat gambar arsitektur RSA pada gambar 3.1

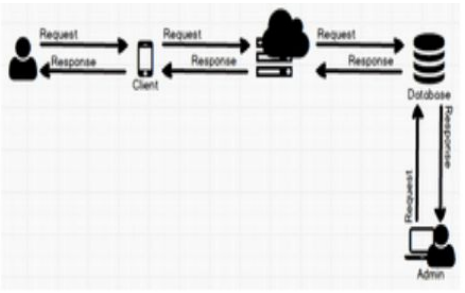


Gambar 2 Arsitektur RSA

3.4 Arsitektur Sistem

Untuk dapat memahami konsep aplikasi yang akan dibangun, dapat melihat gambar arsitektur *system* pada gambar 3.1. pada gambar tersebut telah digambarkan secara garis besar menenai proses dari keseluruhan system.

Gambar 3 Arsitektur Sistem



3.5 Perancangan Basis Data

Basis data dibuat untuk menyimpan informasi atau data yang dihasilkan oleh aplikasi. Berikut merupakan rancangan basis data inti yang diperlukan:

a. Spesifikasi Basis Data

Berikut merupakan spesifikasi dari basis data yang digunakan oleh aplikasi.

1. Nama Tabel : par_user_member
   Isi : Daftar data user client , untuk dapat login pada aplikasi *mobile*
   Primary Key : Kolom email

Tabel 1 Struktur Tabel par_user_member

NAMA KOLOM	TYPE	PANJANG	KETERANGAN
Id	Int	11	Kode pngguna
Email	Varchar	80	Email pengguna
Namadepan	Varchar	80	Nama depan pengguna
Namabelakang	Varchar	80	Nama belakang pengguna
Ktp	Bigint	16	Nomor KTP pengguna
jenis_kelamin	Int	1	Jenis kelamin pengguna
tanggal_lahir	Varchar	12	Tanggal lahir pengguna
Alamat	Text	*255*	Alamat pengguna
Password	varchar	260	Kunci masuk aplikasi
Saldo	Int	12	Saldo pengguna
time_stamp	timestamp	*YYYY-MM-DD HH:ii:ss*	Waktu perubahan data

3.6 Rancangan Layar

a. Tampilan Layar Menu Utama

Tampilan layar Menu Utama pada gambar 4 Pada ditampilan ini pengguna dapat memilih menu pembayaran dan pembelian.

Gambar 4 Tampilan Layar Menu Utama

4 KESIMPULAN

4 Kesimpulan

Kesimpulan yang didapat dari hasil analisa atas masalah dan pemecahannya adalah :

a. Dengan adanya aplikasi BLANJA QuickPay ini dapat memudahkan pengguna setia dari aplikasi keluaran PT METRAPLASA melakukan transaksi pembelian dan pembayaran produk digital

b. Dengan dienkripsinya data transaksi pada sisi *web service*, maka lebih terjaganya keamanan informasi data transaksi pembelian dan pembayaran yang ada dalam aplikasi ini.

c. Data para sisi *web service* yang sudah dienkripsi dapat dikembalikan menjadi informasi asli tanpa mengalami perubahan isi sedikitpun.

d. Data yang disimpan didalam basis data aplikasi ini berupa *ciphertext* yang merupakan hasil enkripsi menggunakan algoritma kriptografi RSA, sehingga pihak yang tidak berwenang tidak dapat mengambil, membaca, atau memanipulasi data tersebut.

e. Berdasarkan dari uji coba penelitian ini didapat hasil nilai rata-rata dari sepuluh kali percobaan pada plaintext 166.6 Byte membutuhkan waktu 7.8 Millisecond pada proses enkripsi dan ciphertext 616.2 Byte membutuhkan waktu 46.4 Millisecond pada proses dekripsi, untuk selisih ukuran antara plaintext dengan chipertext yaitu 449.6 Byte.

4.2 Saran

a. Spesifikasi kebutuhan program harus dipenuhi sehingga aplikasi bekerja dengan baik dan lancar.

b. Perlu dilakukan perawatan/pemeliharaan dan pengawasan dari pihak yang bertanggung jawab terhadap sistem.

c. Melakukan update java pada sistem *web service* secara berkala.

d. Memilih admin yang bertanggung jawab baik dalam penggunaan maupun pemeliharaan program aplikasi.

e. Memberi penjelasan dengan baik kepada pihak terkait tentang program ini, baik dalam mengenkripsi dan mendekripsi, aturan yang berlaku, maupun dalam memanfaatkannya sebagai fasilitas pengaman informasi.

5 DAFTAR PUSTAKA

1. Ariyus, D., 2008. Kriptografi Keamanan Data dan Komunikasi, Graha Ilmu, Yogyakarta.

2. A. Gustavo, dkk., Web Service Concepts, Architectures and Aplications, 2004.

3. Ginting, dkk. 2015., Implementasi Algoritma Kriptografi RSA Untuk Enkripsi Dan Dekripsi Email. Jurnal Teknologi dan Sistem Komputer. Vol,3. (No.2).

4. Heater., 2001. Web Service Cinceotual Arcitecture. IMB Software Group.

5. Munir, R. 2006. Kriptografi. Informatika: Bandung.

6. Menezes, A.J., Oorschot, P.V. & Vanstone, S. 1996. Handbook of Applied Cryptography. CRC Press: New York.

7. Martassari, G, dkk., 2010. Implementasi Web Service Untuk Mendukung Interopabilitas Pada Aplikasi E-Commerce.

8. Muzakir, Rahman. 2107., Ujicoba Sistem Keamanan Informasi Dengan Algoritma Kriptografi RSA dan RSA-CRT pada Sistem e-memo Berbasis Mobile. Jurnal Simetris, Vol, 8 (No.2).

9. Paramartha, A.A Gedhe Yudhi, dkk., 2017. Implementasi Web Service Pada Sistem Pengindeksan dan Pencarian Dokumen Tugas Akhir, Skripsi, Dan Praktik Kerja Lapangan. Jurnal Vol.5, (No.2).

10. Pahrizal, & David Pratama. 2016. Implementasi Algoritma RSA untuk mengamankan data berbentuk teks. Jurnal Pseudocode, Vol.3 (No.1).

11. Paramartha, dkk. 2016., Implementasi Web Service Pada Sistem Pengidendeksan dan Pencarian Dokumen tugas akhir, Skripsi, dan Praktik Kerja Lapangan. Vol.5, (No.2).

12. Qitbiyah, Ulfa Mariatul., 2017. Implementasi JSON Web Service Pada Aplikasi Digital Library Politeknik Sukabumi. Jurnal Vol.2, (No.1).

13. Ruliyanto, dkk. 2015. Implementasi Algoritma RSA untuk Enkripsi dan Dekripasi sms(short Message Service) pada ponsel berbasis android. Jurnal Vol.2 (No.2).

14. Rahmatulloh, dkk., 2018. Keamanan RESTful Web Service Menggunakan JSON Web Token (JWT) HMAC SHA-512. Jurnal Jurnal Vol.7, (No.2).

15. Susanto, & Susilo. 2018., Penerapan Algoritma RSA Untuk Keamanan Data Pada Aplikasi Penjualan CV. Sinergi Computer Lubuklinggau Berbasis Web. Jurnal, Vol.9 (No.2).

16. Tri Rahajoeningroem, Muhammad Aria 2012, Studi Dan Implementasi Algoritma RSA Untuk Pengamanan Data Transkrip Akademik mahasiswa.

17. Wahyadyatmika, dkk. 2014., Implementasi Algoritma Kriptografi RSA Pada Surat Elektronik(Email). Jurnal Vol.,3 (No.4).

18. Zainuzin & Mulyana. 2016. Penerapan Algoritma RSA untuk Keamanan Pesan Instan Pada Perangkat Android. Jurnal Vol. 9, (No.2).