BAB II
LANDASAN TEORI

2.1 Payment Gateway

Payment gateway merupakan sebuah komponen dari infrastruktur penting yang memiliki fungsi untuk memastikan keberlangsungan transaksi tanpa adanya hambatan dan data transaksi pelanggan harus sangat aman ketika melewati jaringan internet [5]. *Payment gateway* dapat dikatakan sebagai jembatan dari dompet elektronik dengan lembaga keuangan. Payment gateway mengarahkan sebuah detail transaksi kepada lembaga keuangan dengan aman dan tidak dapat dimasuki oleh pihak lain. Fungsi dari *payment gateway* sendiri adalah memberikan jalur yang aman dan sudah terenkripsi dengan baik menuju lembaga keuangan. Isi dari data yang terenkripsi berupa detail transaksi (detail kartu pengguna dan informasi transaksi). Sebagai persetujuan transaksi, *payment gateway* akan mengirimkan kembali informasi transaksi kepada dompet digital atau aplikasi kemudian proses tersebut akan melakukan verifikasi untuk melanjutkan pembayaran. Berikut ini merupakan beberapa manfaat dari payment gateway [6]:

1. Kenyamanan transaksi selama 24x7x365.
2. Penggunaan kartu kredit atau debit yang dilakukan secara langsung
3. Jalan transaksi yang efesien dan cepat.
4. Terdapat pilihan pada saat pembayaran.
5. Alur transaksi yang lebih aman dilakukan antara pembeli, penjual, dan institusi finansial.
6. Penggunaan yang fleksibel dikarenakan laporan yang dapat langsung dilihat.
7. Dapat dilakukan dengan menggunakan berbagai jenis mata uang baik dalam bentuk dolar atau rupiah.

8. Memfasilitasi konsumen untuk melakukan pengembalian uang atas transaksi yang dilakukan.
9. Pemilik website atau aplikasi dapat menghilangkan database dikarenakan memakan jumlah besar dan sistem yang rumit dikarenakan banyaknya transaksi yang masuk.
10. Server yang lebih aman dan sudah dapat diuji keamanan server.
11. Servis yang sudah lengkap dan memiliki kontrol administrasi yang sederhana.

2.2 API Gateway

API Gateway dapat memisahkan aplikasi dari pengguna eksternal. Hal tersebut dilakukan dengan cara mengarahkan permintaan dari klien dan aplikasi akhir dengan dinamis ke dalam beberapa aplikasi internal. [7] *API gateway* merupakan servis yang memberikan izin kepada developer untuk membuat, diterapkan, dan mengatur API (*Application Programming Interface*) yang bergerak pada bidang *frontend* yang akan mengakses data dari servis backend seperti akses database dan penyimpanan. *API gateway* berperan sebagai perantara antara klien dan server. *API gateway* menyediakan interface yang seragam dan konsisten sebagai kumpulan dari mikro servis. Hal ini memungkinkan klien dapat berinteraksi dengan developer [8].

2.3 Framework

2.3.1 Echo labstack

Echo / Labstack merupakan sebuah *framework* dalam bahasa pemograman *Golang Framework* memiliki performa yang baik, minimalis, dan dapat diperluas. Hal tersebut membuat penggunaan *framework echo* sesuai pada saat digunakan pada sistem dengan skala yang besar. Echo juga memiliki beberapa keunggulan dalam hal mengoptimalkan router, sudah didukung oleh HTTP/2, data binding,

rendering, *middleware*, dan *error* handling yang Fungsinya dapat dilakukan perubahan sesuai keinginan user. [9]

2.4 Tools

2.4.1 JSON Web Token

JSON Web Token atau dapat disebut sebagai JWT merupakan sebuah token bertipe string JSON yang berukuran padat, informasi bersifat mandiri bertujuan untuk melakukan sistem autentikasi dan pertukaran informasi. JWT memiliki bentuk yang kecil yang dikirimkan dengan menggunakan URL, parameter POST bisa juga di dalam header HTTP, dan dikarenakan memiliki ukuran yang tidak besar maka dapat dikirim dengan cepat. JWT disebut sebagai Informasi yang mandiri dikarenakan isi token yang dihasilkan memiliki informasi dari pengguna. Hal tersebut membuat tidak diperlukannya query pada basis data yang lebih dari satu kali. JWT token akan seperti *password*, sehingga pada saat user berhasil melakukan login maka server akan memberikan sebuah token. Token yang diberikan akan disimpan oleh user pada penyimpanan lokal atau dari *cookies*. Jika *user* ingin mengakses halaman tertentu maka harus memberikan token yang telah diberikan.[10]

Struktur dari payload terdiri dari *header*, *payload*, dan *signature*. Pada gambar 2.2 merupakan gambaran dari struktur JWT token.

Gambar 2.1 Struktur JSON Web Token

Seperti yang sudah dijelaskan sebelumnya pada bagian *header* berisikan authentikasi dari user yang biasa menggunakan algoritma RSA atau HMAC. Kemudian dalam bagian *payload* terdapat informasi dari klien, *token issue date*, informasi *issuer*, dan lainnya. Pada bagian

signature terdapat hash dari *secret key* atau kunci rahasia. Tujuannya untuk memberikan verifikasi integritas dari konten token. [11]

2.4.2 SmartVista

SmartVista merupakan berbagai solusi dan layanan pembayaran yang komperhensif. SmartVista dibangun oleh BPC sebagai *banking technologies*, yang dimana menyediakan solusi pembayaran secara global. SmartVista dirancang untuk memenuhi kebutuhan bank, Lembaga keungan, dan proses pembayaran, yang menyediakan servis untuk mengelolah berbagai layanan pembayaran termaksud manajemen kartu, ATM, pedagang, dan pengalihan pembayaran. SmartVista merupakan sebuah *platform* modular dan dapat diskalakan. Sehingga platform tersebut fleksibel dalam menyediakan solusi pembayaran yang sesuai dengan kebutuhan bisnis tertentu. *Platform* yang disediakan untuk menangani volume transaksi yang tinggi. Hal ini membuat pemrosesan pembayaran menjadi cepat dan aman. SmartVista juga mendukung berbagai saluran pembayaran seperti pembayaran seluler, online, dan *contactless*. [12]

2.4.3 Postman

Postman merupakan sebuah aplikasi untuk melakukan pengujian *Restfull API*. Postman ini akan melakukan pengujian fungsi dari aplikasi dengan response berupa format JSON, XML, dan lainnya. Postman dapat digunakan dalam berbagai parameter yaitu GET, POST, PUT, dan DELETE. Dengan menggunakan Postman memperlihatkan fungsi dari setiap metode dapat berjalan dengan baik atau tidaknya. [13]

2.4.4 JavaScript Object Notation

JavaScript Object Notation (JSON) merupakan sebuah format untuk menukar data dengan ringan. Format ini biasanya digunakan dalam melakukan penukaran data antara klien satu dengan yang lainnya, server dalam aplikasi web. Fungsi JSON adalah sebagai

penyimpanan dan transfer data yang dilakukan secara umum. JSON dibangun dengan tujuan untuk mudah dipahami dan ditulis oleh manusia. Fungsi lainnya juga untuk mudah diproses dan dibangun dengan mesin. Format dalam JSON terdiri dari beberapa pasang kunci nilai. Kunci tersebut dapat berupa string yang biasanya diapit dengan tanda dua tanda petik. Nilai dalam JSON dapat berupa *string*, *boolean*, *null*, *array*, dan objek lain[14].

2.4.5 Docker

Docker merupakan salah satu aplikasi untuk melakukan *deployment* dalam *cloud*. *Container* dibagikan tidak hanya fisiknya tetapi juga *operation system* juga memberikan *support library*. Pada cara *traditional* dengan menggunakan VM (*Virtual Machine*) dengan basis *Hypervisor* hanya memberikan tawaran untuk abstraksi tingkat hardware. Sementara, arsitektur mikro servis memisahkan aplikasi yang kompleks menjadi komponen yang ringan dan saling terhubung. Setiap komponen secara independent dalam menjalankan sebuah mikro servis, dan setiap komponen tersebut dapat diubah dan diubah tanpa mempengaruhi komponen lain. Kehadiran dari *container* dan arsitektur mikro servis ini dapat meningkatkan skalabilitas dan elastisitas aplikasi yang dikembangkan. Sehingga tidak membutuhkan *deploy* satu per satu pada sistem karena dengan menggunakan docker untuk melakukan *deploy* sistem pada beberapa server yang membutuhkan sistem tersebut. [15]

2.4.6 KONG

Kong merupakan lapisan *API gateway* yang bersifat *open source* memiliki skala dan pembangunannya berada di atas Nginx. Kong adalah aplikasi Lua yang berjalan dengan menggunakan Nginx dan didukung dengan Lua-Nginx-Module. Kong mendukung dalam implementasi abstraksi database dan *routing*. Dalam Kong terdapat plugin yang dapat digunakan disesuaikan dengan kebutuhan dari

pengembangan. Contohnya adalah autentikasi dan *logging plugin* yang digunakan untuk menyimpan *log* jalannya aplikasi dengan yang lain.

2.4.7 Vault

Vault merupakan salah satu alat *open source* mengenai *secrets management software package*. Vault ini dibangun oleh Hashicorp kepada bisnis lain untuk menyimpan dan *refresh token* untuk semua pengguna dari berbagai organisasi virtual. Vault sudah mendukung OIDC, kuberos, memiliki plugin arsitektur yang fleksibel dan *rest API*. Sehingga dapat dikatakan bahwa Vault digunakan untuk membuat token baru dan tempat penyimpanannya. Dalam pembangunan token tersebut Vault menggunakan dua algoritma dalam pembuatannya yaitu RSA dan AES. Pembuatan Vault bertujuan untuk keamanan data pada perusahaan besar maupun perusahaan kecil.[16]

2.4.8 Apache Jmeter

Apache Jmeter merupakan software yang bersifat open source. Jmeter merupakan aplikasi Java yang dibangun untuk melakukan tes fungsional dan mengukur kinerja aplikasi. Pada awal dari pembangunan aplikasi Jmeter berfungsi untuk pengujian aplikasi dalam bentuk website, tetapi sekarang diperluas untuk fungsi lain. Jmeter ini digunakan untuk melakukan stress test untuk website, aplikasi FTP, dan *Database server test*. Apache Jmeter juga dapat digunakan untuk menguji kinerja pada sumber daya statis dan dinamis. Hal tersebut bertujuan untuk melakukan simulasi beban berat dari server, jaringan, dan objek. Bertujuan untuk menguji kemampuan untuk menganalisa kinerja secara keseluruhan pada jenis beban yang berbeda. [17]

2.5 Database

Database merupakan sebuah tempat yang memiliki fungsi dalam penyimapanan data. Database juga dapat mengubah, menyimpan, dan

menghapus data [18]. Database digunakan agar ketika perusahaan ingin menggunakan data kembali maka akan lebih mudah karena data telah terstruktur, cepat, dan akurat. Berikut ini merupakan beberapa fungsi penggunaan database [19]:

1. Memudahkan penggunaan dalam melakukan identifikasi pada data dengan cara melakukan pengelompokan data. Dalam satu database terdapat beberapa tabel dan field yang memiliki fungsi berbeda.
2. Mengurangi kemungkinan dalam adanya data ganda.
3. Memudahkan penggunaan dalam melakukan berbagai hal. Database dapat memasukan, menghapus, mengubah data dengan mudah.
4. Menyimpan data dalam bentuk digital dan membutuhkan penggunaan kertas.
5. Menjadi salah satu alternatif mengenai penuyimpanan ruang di suatu sistem.

Terdapat beberapa jenis database. Berikut ini merupakan penjelasan singkat mengenai jenis-jenis database [19]:

1. *Operational Database*

   Operational database dapat mengoperasikan penyimpanan data yang sangat terperinci bertujuan agar dapat lebih mudah digunakan. *Operational database* biasanya dapat digunakan sebagai database pelanggan.

2. *Relational Database*

   Relational database dapat digunakan untuk mencari informasi dalam tabel yang berbeda-beda.

3. *Distributed Database*

   Distributed database dapat digunakan untuk melakukan distribusi data-data yang saling berhubungan dan dapat diakses secara bersama-sama.

4. *External Database*

External database dapat digunakan untuk kebutuhan komersial dikarenakan mudah dalam mengakses database yang di khususkan untuk publik.

Dari database yang telah disebutkan, dalam penelitian ini menggunakan relational database. berikut ini merupakan penjelasan lengkap dari penggunaan relational database.

2.5.1 Relational Database

Relational database merupakan gambaran dari data logik. Relasi dalam database dapat menggambarkan struktur data tanpa melihat seperti apa data tersebut disimpan [20]. Pada manajemen database terdapat beberapa tipe relasi. Berikut ini merupakan macam tipe relasi dalam database [21]:

1. *Degree*

   Terdapat beberapa tipe relasi dalam database yang terlihat dari jumlah *entity*. Berikut ini merupakan penjelasan singkat dari tipe relasi berdasarkan jumlah *entity*:

   a. *Unary relationship* merupakan relasi yang terhubung dengan satu *entity*.

   b. *Binary* relationship merupakan relasi yang terhubung dengan dua *entity*.

   c. *Ternary* relationship merupakan relasi yang terhubung dengan tiga *entity*.

2. *Cardinality*

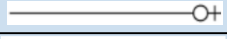
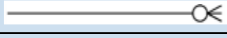
   Dalam relational terdapat tipe relasi yang dilihat dari nilai data pada suatu *entity* yang dapat direlasikan dengan sesamanya. Berikut ini merupakan tipe-tipe jumlah *entity* yang direlasikan [22]:

   a. *One-to-one relationship* merupakan hubungan antara satu data dengan data pada tabel lain memiliki satu hubungan.

b. *One-to-many relationship* merupakan hubungan data dengan data dalam tabel yang berbeda dapat digunakan secara berulang.

c. *Many-to-many relationship* merupakan hubungan banyak data pada satu tabel memiliki hubungan yang memiliki jumlah lebih dari satu data di tabel yang berbeda.

Tabel 2.1 *ERD Symbol Description*

Simbol	Nama simbol
	one
	many
	only one
	zero or one
	one to many
	zero to many

2.5.2 Entity Relationship Diagram

Entity Relationship Diagram atau disebut sebagai ERD merupakan model yang dipakai untuk melakukan desain database. Tujuan pembuatan desain tersebut adalah untuk menggambarkan relasi data yang berada pada database [22]. Setelah pembuatan ERD ini dapat digunakan untuk membuat database yang kemudian akan digunakan dalam aplikasi.

ERD merupakan Teknik yang dipakai untuk membuat model data yang akan dibutuhkan dalam sebuah sistem atau organisasi. Sistem yang membutuhkan ERD biasanya merupakan sistem analisis yang berada pada tahap analisis yang menjadi persyaratan atas projek yang sedang dibangun [23]. Didalam suatu ERD terdapat tiga elemen dasar. Berikut ini merupakan penjelasan singkat mengenai tiga elemen dasar dalam ERD [24]:

1. *Entity*

   Entitas merupakan sebuah objek dari sistem. Biasanya entitas dapat berupa orang, objek, atau event yang didalamnya terdapat informasi yang diperhatikan.

2. *Attribute*

   Attribute merupakan sebuah fitur khusus dari sebuah entitas. Atribut tersebut memberikan sebuah ciri entitas. Tidak memiliki jumlah yang pasti untuk atribut yang berhubungan dengan satu entitas. Atribut yang lebih dari satu memiliki tujuan untuk mengidentifikasi suatu entitas.

3. *Relationship*

   Relasi merupakan hubungan entitas yang satu dengan entitas yang lain.

2.5.3 Structured Query Language

Structured Query Language atau dapat disebut SQL merupakan bahasa pemograman khusus yang dapat digunakan dalam mengelola data dalam relational database. SQL merupakan perintah yang tujuannya adalah untuk mengelolah, memanipulasi, dan menampilkan data yang telah tersimpan dalam database [25]. SQL merupakan *relational database* yang didalamnya terdapat *foreign keys* dan *primary key*. Penyimpanan yang dimiliki SQL dalam bentuk tabel. Menggunakan bahasa *query* yang telah di standarisasi oleh SQL [26].

2.5.4 PostgreSQL

PostgreSQL merupakan salah satu database *open-source* yang memiliki servis yang baik. PostgreSQL menawarkan fitur modern seperti penggunaan query yang kompleks, *Trigger*, *view*, transaksional integritas. Dengan menggunakan PostgreSQL user dapat menambahkan ekstensi untuk tipe data, *function*, *operator*, dan *procedural language*. Akan Tetapi PostgreSQL masih belum terintegrasi dengan data mining untuk menganalisis data [27]. PostgresSQL dapat digunakan pada aplikasi dengan skala menengah ataupun aplikasi sekala besar [28].

2.5.5 NoSQL

NoSQL memiliki makna yaitu "*Not Only SQL*" yang menjelaskan bahwa bukan hanya SQL. NoSQL merupakan jenis database non-relasional hal ini menjadi alternatif dari database SQL yang merupakan database relasional. Perbedaan dari kedua database tersebut terdapat pada skema, SQL yang memiliki skema kaku sedangkan database NoSQL memiliki bentuk skema yang lebih fleksibel dan mudah diubah tanpa mengganggu sistem ketika sedang dijalankan. Dikarenakan database ini memiliki sifat yang flexibel maka NoSQL memiliki skalabilitas yang tinggi sesuai dalam mengelolah data yang besar dan selalu berubah-ubah. [29]

2.5.6 Elasticsearch

Elasticsearch merupakan alat untuk mencari *restful* yang terdistribusi. Elasticsearch didasarkan pada perpustakaan perangkat lunak untuk mengambil informasi. Elasticsearch memiliki perbedaan dari sistem manajemen dengan basis data relasional yaitu RDBMS. Model basis data yang dimiliki Elasticsearch merupakan alat pencari dan menyimpan dokumen dari nilai kunci. Setiap objek dalam Elasticsearch adalah JSON dan oleh karena itu Elastic search tidak menggunakan SQL untuk pembuatan database. Dalam melakukan pencarian dalam Elasticsearch tidak hanya menggunakan query, tetapi juga filter yang berguna pada saat melakukan pencarian dokumen dalam waktu yang cepat. [30]

2.5.7 Kibana

Kibana adalah aplikasi analisis dan visualisasi yang berbasis open-source. Kibana sendiri dibangun dengan tujuan agar dapat bekerja sama dengan Elasticsearch. Kibana dapat mengelola data dalam jumlah besar dan dapat melakukan analisis yang mudah dipahami pengguna. Kibana memberikan fitur penyajian data dalam bentuk diagram, tabel, dan kerangka dalam menerapkan data yang akan dianalisi. Kibana menggunakan fitur *web browser* yang memudahkan dalam

membangun dashboard secara cepat dan realtime dari query Elasticsearch. [31]

2.6 Algoritma

2.6.1 Rivest Shamir Adleman (RSA)

Rivest Shamir Adleman (RSA) merupakan salah satu sistem enkripsi yang menggunakan mekanisme enkripsi asimetris. Enkripsi asimetris ini menggunakan dua kunci untuk publik dan privat[32]. Berikut ini merupakan detail dari proses dari pembuatan kunci dengan menggunakan RSA [7]:

1. Memilih dua bilangan prima secara random. Asumsikan variabel tersebut p dan q.
2. Kemudian hitung hasil dari rumus berikut:
$$n = p \,.\, q$$
3. Hitunglah $\Phi(\mathrm{n})$ dengan rumus berikut:
$$\Phi(\mathrm{n}) = (p-1).(q-1)$$
4. Kemudian pilih kunci publik. Asumsikan sebagai variabel e. variabel e dipilih memiliki dua kriteria yaitu hari diantara nilai 1 serta $\Phi(\mathrm{n})$ dan kemudian bilangan prima yang tidak memiliki relatif dengan $\Phi(\mathrm{n})$ dan n.
5. Kemudian memilih private key, dengan memberikan asumsi sebagai variabel d. Variabel d dicari dengan menggunakan rumus sebagai berikut:
$$d.e\big(mod\ \Phi(\mathrm{n})\big) = 1$$
Dimana mencari variabel d dengan rumus sebelumnya dengan hasil 1.

Dari rumus tersebut didapatkan dua kunci (e, n) sebagai *public key* dan d sebagai *private key*. Berikut ini merupakan rumus encryption dalam algoritma RSA:

$$h(m,k) = m^k\ mod\ 100$$

Dari rumus tersebut didapatkan dua kunci (e, n) sebagai *public key* dan d sebagai *private key*. Berikut ini merupakan penjelasan fase enkripsi untuk plain text[33]:

1. Memilih pesan yang ingin di enkripsi, asumsikan pesan menjadi variabel m
2. Orang yang mengirimkan pesan akan mendapat variabel e dan n yang merupakan public key dan menerima rumus untuk encryption.
3. Memilih angka random yang didefinisikan menjadi variabel k, dimana k merupakan angka antara 1 sampai n-1
4. Kemudian melakukan kalkulasi variabel A dengan rumus:

$$A = k^e \ mod \ n$$

5. Kemudian kalkulasikan variabel B dengan rumus:

$$B = m \ . (k + 1)^e \ mod \ n$$

6. Kalkulasikan variabel H dengan rumus:

$$H = h(m, k)$$

7. Kemudian kirimkan cyphertext C = (A, B, H) kepada sistem untuk dekripsi

Berikut ini merupakan penjelasan fase decrypt untuk melihar isi dari plain text[33]:

1. Sistem akan menerima pesan berupa C = (A, B, H) dari orang yang mengirimkan pesan.
2. Sistem akan mengkalkulasi variabel k dengan rumus:

$$k = A^d \ mod \ n$$

3. Sistem akan mengkalkulasi variabel m dengan rumus:

$$m = B \ /(k + 1)^e \ mod \ n$$

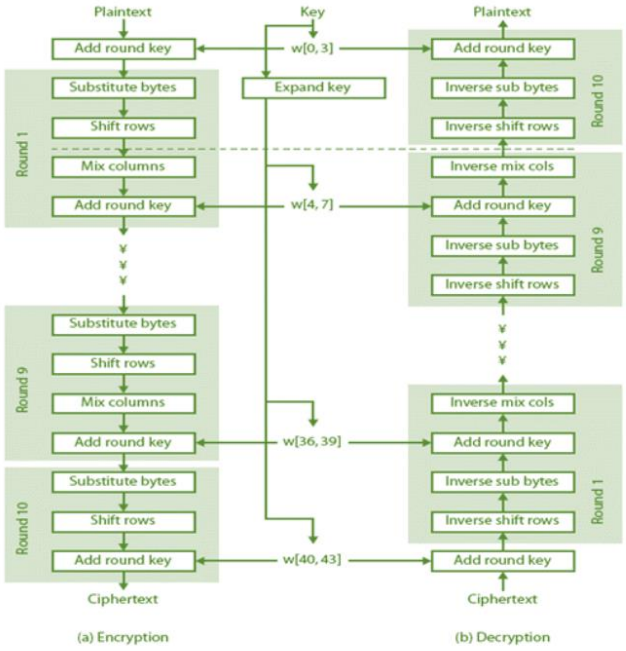
4. Sistem akan mengkalkulasi variabel H dengan rumus:

$$H = h(m, k)$$

2.6.2 Advanced Encryption Standard (AES)

Algoritma *Advanced Encryption Standard* (AES) merupakan algoritma yang sering digunakan untuk melakukan enkripsi data. Algoritma AES menggunakan mekanisme enkripsi simetris. AES mendukung beberapa besar kunci enkripsi yaitu 128, 192, atau 256 bit. semakin besar byte akan semakin banyak perulangannya, yaitu 10, 12, dan 14 kali (128, 192, dan 256) [34].

Algoritma AES akan melakukan enkripsi blok data dengan cara menjalankan beberapa tahap secara berulang. Dalam enkripsi ini menggunakan 16 Byte data blok yang diorganisir menjadi blok 4 x 4 dalam bentuk matrix. Setiap sel dari blok tersebut terdiri dari 1Byte. Pada gambar 2.1 merupakan struktur jalannya algoritma AES [35]:



Gambar 2.2 Basic Structure of AES

Berikut ini merupakan penjelasan mengenai tahap yang terdapat pada algoritma AES [36]:

1) *Add Round Key*

   Setiap sel blok data akan dilakukan XOR dengan *key block*. Round key didapatkan dari blok kunci yang khusus untuk

putaran yang didapatkan dari kunci enkripsi dan nomor putaran.

2) *Sub Bytes*

Setiap byte dalam blok akan diubah dengan menggunakan tabel *hexadecimal multiplication*.

3) *Shift Rows*

Setiap baris dalam blok data akan digeser secara tersusun berdasarkan 0, 1, 2, atau 3 dari lokasi mereka.

4) *Mix Column*

Pada bagian ini merupakan proses perkalian dari setiap kolom blok data, yang dimana setiap kolom data matrix dikali dengan matrix tetap yang berbentuk 4 x 4.

2.6.2.1 Key Scheduling

Key scheduling merupakan bagian dari algoritma AES. *Key scheduling* digunakan untuk membuat kunci untuk setiap putaran. Simbol dari kunci ini disebut K^i yang artinya kunci ke i. Untuk kunci ke 0 (pertama) merupakan kunci inputan dari user. Dari kunci ke 0 akan membuat kunci lain untuk setiap putaran setelah putaran pertama. Untuk kunci AES-256 memiliki kunci sebesar 256 bit. *Key scheduling* membagi kunci menjadi 128-bit menjadi 4 kolom 4 byte (4 x 4). Pada putaran ke 15 kolom pertama sebesar 4 byte menyebar ke kolom berikutnya dan kemudian menghasilkan kolom kedua dan prosesnya akan mengulang. Dalam pembuatan kunci baru akan melewati beberapa proses yaitu RotWord, SubWord, dan operasi XOR. Kemudian kolom keempat akan menyebar kekolom pertama pada putaran berikutnya. Berikut ini merupakan langkah dari pembuatan kunci di setiap round [37]:

1. RotWord: operasi rotasi siklik. Sebagai contoh, input empat byte (x_1, x_2, x_3, x_4) menghasilkan output empat byte (x_2, x_3, x_4, x_1).
2. SubWord: substitusi byte di mana setiap byte input digantikan dengan byte lain sesuai dengan permutasi non-linear pada kotak S-Box.
3. Rcon: sebuah array kata konstan putaran, yang didefinisikan sebagai Rcon = (xi, {00}, {00}, {00}). Pada gambar merupakan gambaran dari *matrix round constant*.

Round	1	2	3	4	5	6	7	8	9	10
Rcon	01	02	04	08	10	20	40	80	1b	36
	00	00	00	00	00	00	00	00	00	00
	00	00	00	00	00	00	00	00	00	00
	00	00	00	00	00	00	00	00	00	00

Kemudian akan kita lakukan XOR dengan menggunakan rumus dibawah.

$$g(w_3) = SubWord(w_3) \oplus R1$$

Untuk menemukan $g(w_3)$, w_3 yang sudah di SubWord dan R1 atau Rcon(1) sebelumnya akan dijadikan *binary* kemudian akan dilakukan XOR. Hasil XOR akan dikembalikan menjadi *hexadecimal*. Hasil yang didapatkan menjadi kunci untuk round selanjutnya.

2.7 Use Case Diagram

Use case diagram merupakan model dari sistem *view* pada bagian pengamat *external*. *Use case diagram* dapat dikatakan sebagai *static view* dari sebuah sistem. *Use case diagram* untuk menggambarkan sistem berdasarkan Bagaimana sistem tersebut bekerja. Skenario merupakan hasil dari aktor yang berinteraksi dengan sistem. *Use case diagram* membantu dalam menentukan fitur dari sebuah sistem. Dengan menggunakan *use case diagram* dapat membantu kemungkinan informasi baru berdasarkan skenario yang sudah ada. Contohnya adalah dari *testing* dengan membuat skenario dari *use case diagram* dapat membangun *test case* dan komunikasi dengan klien dengan menyediakan memberikan gambaran apa yang akan dilakukan sistem. [38]

2.8 Software Development Life Cycle

Software development life cycle atau dapat disebut sebagai SDLC merupakan suatu metode yang berguna dalam mendesain, membangun, dan memonitor informasi dan sistem dalam perusahaan [39]. Dalam SDLC terdapat beberapa fase yaitu sebagai berikut [40]:

1. *Planning*

   Pada tahap *planning* kelayakan dari sistem akan diperhatikan. Bagian yang di perhatikan adalah dari segi biaya, teknikal, dan operational. Berikut ini beberapa aktivitas pada tahap planning:

   a. Mengidentifikasi masalah yang akan diselesaikan dari sistem yang akan dibangun dan objek yang berkaitan dengan sistem.
   b. Menentukan nilai bisnis yang akan dikeluarkan dalam bentuk pengeluaran yang direncanakan. Pengeluaran akan dilihat dari biaya untuk pembuatan sistem dan jumlah pendapatan setelah sistem dibangun.
   c. Membuat rencana kerja, mengalokasikan waktu kerja, dan penggunaan sumber daya.
   d. Mengorganisir team dalam projek
   e. Mengontrol dan mengarahkan projek

2. *Analysis*

   Pada tahap analysis akan menyelidiki sistem yang sedang berjalan. Hal tersebut bertujuan agar memperoleh dokumentasi yang lengkap mengenai kebutuhan pengguna dan *scope* dari sistem. Dalam tahap analisis juga bertujuan untuk mendapatkan cara dalam membangun sistem baru. Berikut ini merupakan cara yang akan dilakukan dalam tahap analisis:

   a. Melakukan analisis mengenai kebutuhan informasi, hardware, dan software yang akan digunakan dalam pembuatan sistem. Biasanya akan dilakukan dalam dokumentasi spesifik yang biasanya disebut sebagai *Software Requirement Specification* (SRS).

b. Dalam tahap ini akan melakukan analisis *scope* dari sistem yang akan dibangun.

3. *Design*

   Pada tahap design alur kerja akan dijelaskan dan desain dari sistem akan dibuat disesuaikan dengan analisis yang sudah dibuat sebelumnya. Pada tahap ini akan mengahasilkan sistem *prototype*, yang merupakan *design system* yang sudah siap untuk dibangun dan dikerjakan:

   a. Desain database.
   b. Desain grafis.
   c. Desain program.
   d. Desain *system security.*
   e. Kompilasi dari proposal sistem yang akan dibuat.

4. *Development*

   Pada tahap *development*, pengembangan dari sistem *prototype* yang telah dibuat sebelumnya akan dieksekusi. Hasil dari tahap ini berupa sistem yang sudah jadi. Tetapi biasanya akan berupa aplikasi desktop atau website. Kegiatan yang akan dilakukan pada tahap ini adalah menulis program dan mengeksekusi program, pemograman basis data, dan pembuatan *interface*.

5. *Testing*

   Dalam tahap testing dalam sistem *software* yang telah dibangun. Akan melakukan percobaan apakah sistem dapat berjalan dengan optimal atau belum. Jika terjadi masalah dalam sistem saat dijalankan, maka sistem akan dilakukan perbaikan untuk mendapatkan hasil yang lebih baik dan aplikasi dapat digunakan.

6. *Implementation*

   Pada tahap implementasi sistem yang telah dibuat telah lulus saat pengujian *testing* dan akan dijalankan pada *user* akhir. Aktivitas yang akan dilakukan adalah melakukan instalisasi sistem *user*.

7. *Maintenance*

Pada tahap *maintenance* ini merupakan kelanjutan dari tahap implementasi yang akan dilakukan beberapa kali. Tujuannya adalah performa sistem tetap baik.

Dalam SDLC memiliki beberapa model diantaranya adalah model prototyping. Berikut ini merupakan penjelasan mengenai model prototyping.

2.8.1 Agile

Agile merupakan suatu metode yang memiliki respon yang *flexible* jika terjadi perubahan dan dapat meningkatkan kualitas software dalam pengembangan projek. Metode Agile menggambarkan filosofi yang berdasarkan kolaborasi antara orang dan memberikan value yang dicari oleh klien agar puas dengan progres pengembangan sistem dari setiap tahap pengerjaan. Terdapat empat dasar dari metode Agile, yaitu [45]:

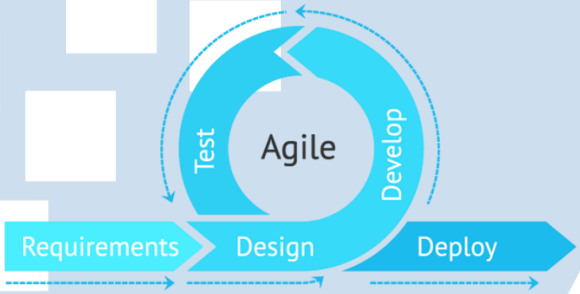
1. Individual dan interaksi mengenai proses dan alat
2. Mengerjakan pengembangan software berdasarkan dokumentasi yang komperhensif
3. Membutuhkan kolaborasi klien sebagai negosiasi kontrak pada setiap tahapannya.
4. Memberikan respon untuk mengubah rencana kerja

Terdapat beberapa keuntungan dalam menggunakan metode Agile dalam mengembangkan sistem software, yaitu sebagai berikut[45]:

1. Mengurangi biaya
2. Mengurangi waktu untuk instalasi dan maintenance karena projek sudah dipahami klien
3. Semua mengenai pembangunan sistem telah diketahui klien dari awal
4. Meningkatkan kualitas *software*
5. Prioritas management yang lebih baik
6. Memberikan komunikasi dan rencana yang baik antara IT dan bisnis
7. Mengurangi resiko

8. Meningkatkan kerja sama tim
9. Meningkatkan produktifitas
10. Memudahkan proses pengembangan
11. Meningkatkan kedisiplinan karyawan

Berikut ini merupakan siklus dari metode pengembangan dengan menggunakan Agile[46]:



Gambar 2.3 *Agile Method Diagram*

Bada gambar 2.3 merupakan siklus pada metode Agile. Dalam metode Agile terdiri dari beberapa langkah. Berikut ini merupakan dari setiap langkah pada Agile [46]:

1. *Requirements*

   Pada awalnya dalam membangun program akan dilakukan pengumpulan *requirement*. *Requirement* tersebut akan dianalisa dan ditentukan. *Requirement* ini harus melibatkan kebutuhan dari *user*.
2. *Design*

   Setelah mengumpulkan *requirement* yang dibutuhkan maka akan melakukan desain sistem. Desain sistem dapat berupa perancangan sistem yang dibangun. Bagian ini melibatkan arsitektur dan fitur yang dimiliki sistem. Dalam fase ini akan dibangun ilustrasi dari sistem yang dibuat.
3. *Develop*

Dalam fase ini akan mengimplementasikan perancangan yang sebelumnya dilakukan menjadi program. Program yang dibangun akan disesuaikan dengan kebutuhan *user*.

4. *Testing*

   Setelah program selesai dibangun maka akan melakukan *testing*. Pada awalnya testing akan di test oleh *developer*. Setelah *developer* melakukan *test* maka akan diberikan kepada *user*. *User* akan memberikan *review* kepada pengembang sistem.

5. *Deploy*

   Melakukan implementasi program pada server untuk digunakan sesuai dengan tujuan awal.

2.8.2 UAT

User Acceptance Testing (UAT) merupakan sebuah proses yang memverifikasi mengenai solusi yang dibuat dalam sistem sudah sesuai dengan keinginan pengguna atau masih terdapat kekurangan. Proses ini memiliki perbedaan dengan menguji sistem, dikarenakan dalam proses ini memastikan bahwa sistem tidak mengalami kerusakan pada saat berjalan dan telah sesuai dengan keinginan pengguna serta memastikan solusi yang diberikan telah bekerja dengan baik untuk pengguna.

UAT pada umumnya dilakukan oleh klien atau pengguna akhir, biasanya lebih focus pada identifikasi masalah sederhana seperti kesalahan ejaan, kerusakaan pada saat menjalankan sistem. Penguji dan pengembang akan mengidentifikasi serta memperbaiki masalah tersebut selama tahap pengujian fungsionalitas. Setelah UAT telah dilakukan akan melakukan perhitungan presentase. Berikut ini merupakan perhitungan presentase UAT [48].

$$\frac{Total\ Skor}{Total\ keseluruhan\ UAT}\ x\ 100\%$$

2.9 Unified Modeling Language

Unified Modeling Language singkatan dari UML yang merupakan model yang memiliki tujuan untuk merancang dan mengembangkan sistem yang berbasis *object-oriented*. UML memberikan standard penulisan yang dapat disebut sebagai blueprint untuk perancangan sistem. *Blueprint* dari rancangan ini terdiri dari proses bisnis, penulisan kelas-kelas dalam bahasa spesifik, skema dari database, dan komponen yang dibutuhkan sistem. Dalam UML terdiri dari beberapa tahap untuk mendesain sebuah sistem yaitu *use case diagram*, *activity diagram*, *class diagram*, dan *database design*. [49]

2.9.1 Use Case Diagram

Diagram *use case* merupakan bagian dari UML yang memiliki tujuan untuk mengartikan fungsionalitas dan grafis dari sistem berdasarkan aktor, aktivitas, beserta dengan hubungannya. Dalam use case juga berfungsi untuk menggambarkan interaksi antara satu atau lebih aktor dengan sistem yang ingin dirancang. Yang dimaksud dengan aktor adalah entitas yang akan berinteraksi dengan sistem. Kemudian terdapat aktivitas atau use case yang untuk menunjukan aktivitas apa saja yang dilakukan dalam sistem. Kemudian terdapat garis penghubung yang disebut *association* antara aktor dan use case untuk menggambarkan apa yang dilakukan aktor pada sistem. Terdaoat garis yaitu Extend yang menggambarkan suatu aktivitas ditambahkan dengan aktivitas lain. kemudian terdapat garis penghubung include yang menggambarkan hubungan aktivitas bergantung dengan aktivitas lain. [49]

2.9.2 Activity Diagram

Activity diagram dibuat untuk memahami lebih dalam mengenai sistem yang akan dibuat. Diagram ini dibuat untuk menggambarkan proses aktivitas yang sebelumnya telah digambarkan di use case diagram. Pada bagian ini menggambarkan lebih detail jalannya suatu aktivitas. Dalam activity diagram ini terdiri dari status awal, aktivitas,

percabangan, dan status akhir. Status awal digunakan untuk pembukaan aktivitas sebelum dimulai. Aktivitas merupakan apa yang dilakukan dalam sistem. Percabangan merupakan pilihan proses ketika terdapat beberapa kemungkinan proses. Kemudian status akhir yang merupakan akhir dari aktivitas pada sistem. [49]

2.9.3 Class Diagram

Class diagram merupakan model yang menggambarkan sistem yang menggunakan kelas atau abstraksi. Dalam class diagram terdiri dari kelas-kelas dan hubungan antara kelas tersebut. Diagram ini merupakan kelanjutan dari activity diagram. Diagram ini digunakan untuk menunjukan peran dan tanggung jawab umum mengenai entitas yang menjelaskan apa yang dilakukan sistem. Diagram ini digunakan untuk menggambarkan arsitektur yang didapat dari struktur kelas. Dalam class diagram terdiri dari class, association, generalization, composition, dan aggregation. Class merupakan kelas yang ada pada struktur di sistem. Association merupakan relasi yang menghubungkan suatu kelas dengan kelas lain. Generalization merupakan relasi yang menghubungkan satu kelas dengan kelas lain dengan makna khusus dan umum. Composition merupakan relasi yang memiliki makna kebergantungan dengan kelas lain yang berhubungan. Aggregation merupakan relasi kelas dengan kelas lain yang menggambarkan makna semua bagian. Pada relasi aggregation kelas yang berhubungan dapat berdiri sendiri tanpa kelas lain yang memiliki hubungan langsung. [49]

2.10 Bahasa Pemograman

2.10.1 Golang

Golang atau disebut Go merupakan salah satu bahasa pemograman open-source yang memiliki kemiripan dengan bahasa pemograman C dan C++ yang dirilis oleh Google. Go memiliki fitur yang sangat baik sehingga bahasanya cukup simple. Oleh karena itu, mudah untuk dipelajari, sintak yang ringan dan manajemen memori

yang baik. Kelebihan lain dalam menggunakan Golang juga memiliki waktu eksekusi yang cepat dan terstruktur. Adanya static typing membuat go menjadi bahasa pemograman yang aman saat digunakan. Go juga memiliki dokumentasi yang lengkap dan memberikan dukungan cross compilation. [50]

Tabel 2.2 Tabel Penelitian Terdahulu

Nama Artikel dan Peneliti	Journal	Hasil	Yang Diadopsi
Judul: *Enhancing the Security of Online Card Payment System.* Y **Penulis:** Srivani Bobba dan Renu Deepti Surapaneni	International Journal of Advanced Trends in Computer Science and Engineering	Hasil dari penelitian ini bertujuan untuk menciptakan sistem pembayaran online yang aman dengan menggunakan Algoritma AES dan RSA. Sistem ini dibuat dengan tujuan menghindari serangan yang kemungkinan terjadi. Hal tersebut dikarenakan serangan tersebut merusak fungsi dari sistem pembayaran yang melakukan verifikasi detail dari nomor kartu. Dari penyerangan tersebut membantu untuk mendapatkan informasi keamanan yang dibutuhkan untuk melakukan transaksi online. Keamanan tersebut dapat ditingkatkan dengan menggunakan implementasi algoritma yang dikembangkan dalam penelitian ini	Dalam penelitian ini akan mengadopsi penggunaan algoritma RSA dan AES dalam melakukan pengembangan sistem payment gateway pada PT. XYZ.

Nama Artikel dan Peneliti	Journal	Hasil	Yang Diadopsi
		dalam *payment gateway*.	
Judul: Rancang Bangun *Payment Gateway* Pada E-Commerce Berbasis Syariah **Penulis:** Mohammad Izzi Fajrin, Irwan Alnarus Kautsar, Sukma Aji	Procedia of Engineering and Life Science	Hasil dari penelitian ini bertujuan dalam membangun aplikasi yang disediakan untuk para turis memesan tempat wisata secara online dan melakukan pembayaran tiket secara online. Untuk bagian *payment gateway* akan menghasilkan output berupa JSON untuk membuktikan transaksi tersebut sukses. Pembangunan sistem dengan menggunakan bahasa pemograman Python, framework Flask dan Postman untuk pengujiannya. Dalam pembangunan aplikasinya menggunakan metode Agile.	Dalam penelitian ini akan mengadopsi penggunaan Postman untuk pengujian aplikasi dalam pengembangan sistem *payment gateway* pada PT. XYZ.
Judul: *Promize - Blockchain and Self Sovereign Identity Empowered Mobile ATM Platform* **Penulis:** Eranga Bandara, Xueping Liang,	Intelligent Computing	Hasil dari penelitian ini adalah membangun sistem *peer-to-peer* yang berbasis blockchain yaitu Promize. Promize bertujuan untuk meminimalkan biaya dalam melakukan peer-to-peer sebagai	Dalam penelitian ini akan mengadopsi penggunaan bahasa pemograman Golang dan Docker sebagai container pada server. Kemudian akan diimplementasikan juga user authentication

Nama Artikel dan Peneliti	Journal	Hasil	Yang Diadopsi
Peter Foytik, Sachin Shetty, Nalin Ranasinghe, Kasun De Zoysa, dan Wee Keong Ng		alternatif untuk sistem ATM dan debit / kredit tradisional. Sistem tersebut dibangun agar pengguna dapat melakukan penarikan uang dari orang sekitar dan dotoritas perbankan yang telah terdaftar.	dengan menggunakan JWT auth Token dalam pengembangan sistem payment gateway pada PT. XYZ.
Judul: Implementasi Keamanan pada Transaksi Data Menggunakan Sertifikat Digital X.509 **Penulis:** Is Mardianto, Kuswandi	Ultimatics: Jurnal Teknik Informatika	Hasil dari penelitian ini bertujuan untuk mengimplementasi teknologi yang memiliki sertifikat digital X.509. Tujuannya adalah untuk memberikan layanan mobile web servis pada kliennya. Dalam mengamankan transaksi pertukaran layanan web dan ponsel akan menggunakan algoritma SHA, Diffie-Helman, dan *Advanced Encryption Standard* (AES). Algoritma SHA akan digunakan untuk otentikasi	Dalam penelitian ini akan mengadopsi penggunaan algoritma dalam melindungi terutama AES transaksi yang dilakukam untuk melindungi data user.

Nama Artikel dan Peneliti	Journal	Hasil	Yang Diadopsi
		pengguna, Algoritma Diffie-Helman akan digunakan untuk kunci publi, dan AES untuk kriptografi simetris.	
Judul: Pengembangan Sistem Informasi Public E-Marketplace pada PT XYZ **Penulis:** Agus Salaiman	Jurnal ULTIMA InfoSys	Dalam penelitian ini bertujuan untuk meningkatkan jumlah turis yang masuk ke Belitung terutama di Tanjung Kelayang yang merupakan salah satu letak spesial ekonomi, dengan menyediakan layanan teknologi berdasarkan kebutuhan fasilitas yang dibutuhkan untuk tour.	Dalam penelitian ini akan mengadopsi penggunaan metode SDLC dengan menggunakan model Agile sebagai alur pembangunan sistem payment gateway pada PT. XYZ

2.11 Penelitian Terdahulu

Pada tabel 2.2 merupakan penelitian terdahulu yang dijadikan acuan pada penelitian ini. Pada penelitian pertama membahas mengenai penggunaan algoritma RSA, AES, dan SHA untuk meningkatkan keamanan pembayaran online [51]. Dalam Penelitian tersebut bertujuan untuk menghindari serangan yang kemungkinan besar terjadi pada *payment gateway*. Serangan yang dimaksud menyebabkan rusaknya fungsi dari sistem *payment* yang akan mengambil verifikasi detail dari nomor kartu. Dari data tersebut membantu penyerangan untuk mendapatkan informasi keamanan yang dibutuhkan untuk melakukan transaksi online. Untuk memperkuat keamanan pada saat melakukan transaksi, sistem harus ditingkatkan dengan menggunakan algoritma yaitu AES, RSA, dan SHA. Kemudian pada penelitian ke-empat membahas mengenai keamanan bertransaksi yang menggunakan sertifikat digital X.509 [52]. Dalam penelitian ini akan menggunakan algoritma SHA,

Diffie-Helman, dan *Advanced Encryption Standard* (AES). Algoritma SHA akan digunakan untuk otentikasi pengguna, Algoritma Diffie-Helman akan digunakan untuk kunci publik, dan AES untuk kriptografi. Penelitian ini akan dijadikan referensi dalam penggunaan algoritma AES dan RSA.

Dalam penelitian terdahulu ke-dua membahas mengenai pembangunan aplikasi yang disediakan untuk para turis yang berada di Indonesia untuk dapat memesan dan membayar tiket secara online untuk memesan tiket tempat wisata [53]. Untuk bagian pembayaran hasil yang akan dihasilkan berupa response berhasil atau tidaknya transaksi yang berbentuk JSON. Dalam pengujian tersebut juga menggunakan Postman untuk melihat hasil akhir status transaksi yang berjalan. Penelitina ini juga akan menjadi referensi penggunaan Postman sebagai alat pengujian dan JSON.

Penelitian terdahulu ke-tiga membahas mengenai pembangunan sistem *peer-to-peer* yang berbasis blockchain dinamakan Promize [54]. Promize bertujuan untuk meminimalkan biaya dalam melakukan peer-to-peer selain menggunakan debit atau kredit tradisional. agar penarikan uang dapat dilakukan dimanapun terutama pada otoritas perbankan yang telah mendaftar (Indomaret, alfamart, dan toko lain) dan dengan pengguna lain. Dalam pembangunan *payment gateway* pada penelitian tersebut menggunakan bahasa pemograman Golang dan Docker sebagai container untuk deployment dalam server. Kemudian dalam penelitian tersebut menggunakan JWT authentication sebagai hak akses pengguna yang mengakses. Penelitian ini akan menjadi referensi dalam pengembangan menggunakan bahasa pemograman Golang, Docker, dan JWT sebagai authentication user yang akan melakukan pembayaran.

Penelitian terdahulu ke-lima akan membahas pembangunan aplikasi untuk turis yang akan data ke Belitung terutama pada wilayah Tanjung Kelayang yang merupakan wilayah ekonomi [55]. Pengembangan aplikasi tersebut dengan menyediakan fasilitas untuk servis tempat wisata di daerah Belitung dengan menggunakan Android Studio. Aplikasi yang dibuat menggunakan

metode pengembangan SDLC berupa Agile. Sehingga dalam penelitian ini akan mengadaptasi metode pengembangnya dengan menggunakan Agile.