BAB II

LANDASAN TEORI

2.1 Perangkat Lunak / *Software*

Berikut ini merupakan definisi perangkat lunak / *software* menurut beberapa ahli, antara lain:

1. Menurut Roger S. Pressman, perangkat lunak adalah suatu perintah program yang teradapat di dalam sebuah komputer [1].
2. Menurut Melwin Syafrizal Daulay, *software* adalah sebuah perangkat yang berfungsi sebagai pengatur aktivitas kerja komputer dan seluruh intruksi yang mengarah pada sistem komputer [2].
3. Menurut Adam Augustyn, *software* adalah kumpulan instruksi yang memberikan perintah kepada komputer tentang apa yang harus dikerjakan [3].

Berdasarkan informasi yang dikemukakan oleh para ahli diatas, dapat disimpulkan bahwa *software* adalah sebuah perangkat lunak pada komputer yang diciptakan oleh manusia dengan tujuan membantu serta meringankan pekerjaan manusia yang bersifat *sequential* . Sebagai contoh

yaitu aplikasi *Point of Sales (POS)* untuk mendukung kebutuhan operasional pada sebuah bisnis.

2.2 *Sistem Point of Sales (POS)*

Dilansir dari *website* glints.com, *point of sales* adalah sebuah sistem yang berfungsi untuk membantu dan memudahkan para pemilik bisnis dalam mengontrol kegiatan bisnis dalam waktu tertentu [4].

Berdasarkan informasi yang diperoleh dari *website* glints.com, dapat disimpulkan bahwa *point of sales* adalah sebuah sistem yang digunakan untuk mendukung kegiatan operasional suatu bisnis. Sistem *point of sales* berfungsi untuk mencatat setiap transaksi bisnis yang terjadi antara penjual dan pembeli, biasanya sistem *point of sales* memberikan pengetahuan kepada *user* mengenai sisa stok, penjualan, pengeluaran, dan informasi-informasi pendukung lainnya. Pada umumnya sistem *point of sales* berbasis *desktop* ataupun *mobile* (tablet, *smartphone*, dan lainnya).

2.3 Aplikasi Berbasis Android

Menurut Kristijan Lucic, aplikasi berbasis android adalah bagian dari perangkat lunak yang dapat digunakan pada sistem operasi android [5]

Berdasarkan informasi yang dikemukakan oleh para ahli diatas, dapat disimpulkan bahwa aplikasi berbasis android adalah sebuah *software* yang dirancang khusus untuk berjalan diatas perangkat yang hanya memiliki

sistem operasi android dan berfungsi untuk mendukung kebutuhan manusia pada periode waktu yang bervariasi. Saat ini, berbagai macam aplikasi berbasis android tersebar dengan berbagai macam kegunaan di berbagai macam *platform* seperti *playstore* dan *platform* lainnya. Aplikasi berbasis android menjadi sangat cepat berkembang dan digunakan oleh banyak penggguna karena *smartphone* yang digunakan oleh pengguna sehari-hari menggunakan sistem oprasi yang sama seperti aplikasi yang ada yaitu "android". Tentunya hal ini menawarkan efisiensi dan kemudahan dalam mengakses sebuah aplikasi.

2.4 Basis Data

Dilansir dari britannica.com, *database* adalah kumpulan data ataupun informasi yang diatur secara khusus untuk pencarian dan pengambilan cepat oleh komputer. *Database* disusun untuk memfasilitasi penyimpanan, pengambilan, modifikasi, dan penghapusan data sehubungan dengan berbagai operasi pemrosesan data [6].

Berdasarkan informasi diatas, dapat disimpulkan bahwa *database* adalah sebuah gudang penyimpanan data yang berisikan kumpulan data dari satu sistem atau lebih. Kumpulan data yang ada di dalam *database* disimpan secara sistematis sehingga dapat diolah ataupun dimanipulasi untuk menghasilkan sebuah informasi kepada pengguna. Basis data biasanya dimiliki setiap aplikasi untuk menyimpan data-data yang diperlukan

diwaktu yang akan datang dengan tujuan tertentu. Selain itu, basis data tidak hanya digunakan untuk menyimpan data namun basis data juga memiliki fungsi-fungsi lain seperti automasi menggunakan fitur *trigger*, meyimpan logika pemrograman menggunakan *stored procedure* atau *function*, dan masih banyak lagi.

2.5 *Stored Procedure*

Berikut ini merupakan definisi *stored procedure* pada basis data menurut beberapa ahli, antara lain:

1. Menurut Ranga Babu, *stored procedure* adalah kumpulan pernyataan yang dikelompokkan sebagai unit logis dan disimpan didalam database [7].

2. Menurut Emil Drkusic, *stored procedure* adalah salah satu objek database yang kuat pada *database*. *Stored procedure* dapat membantu menangani banyak tugas serta meningkatkan kinerja dan keamanan [8].

Berdasarkan informasi yang dikemukakan oleh para ahli diatas, dapat disimpulkan bahwa *stored procedure* pada *database* adalah sebuah kode pemrograman SQL yang ada berada di dalam *database*, berfungsi untuk melakukan kegiatan yang bersifat *sequential* / berulang. Penggunaan *stored procedure* dapat membantu seorang *programmer* dalam melakukan efisiensi waktu pada waktu penulisan kode SQL yang sifatnya berulang.

Selain itu, *stored procedure* juga dapat mempercepat, mengamankan, serta meringankan beban aplikasi pada saat proses transaksi berlangsung.

2.6 *Function*

Dilansir dari meego.id, *function* pada basis data adalah salah satu fitur di SQL, berupa kumpulan perintah SQL yang disimpan di dalam *database* [9].

Berdasarkan informasi yang dikemukakan oleh situs meego.id, dapat disimpulkan bahwa *function* pada *database* adalah salah satu fitur SQL, berupa kumpulan perintah SQL yang disimpan di dalam *database*. Umumnya, *function* dikombinasikan dengan *stored procedure* dalam penggunaannya.

2.7 *Trigger*

Berikut ini merupakan definisi *trigger* pada basis data menurut beberapa ahli, antara lain:

1. Menurut Wenzel Kris, *trigger* pada basis data adalah *stored procedure* khusus yang dijalankan saat tindakan tertentu terjadi dalam database. Sebagian besar *trigger* ditentukan untuk dijalankan saat perubahan dilakukan pada tabel-tabel tertentu. *Trigger* dapat melakukan tindakan *Data*

Manipulation Language (*DML*) seperti *insert*, *update*, dan *delete* [10].

2. Menurut Dedianto, *trigger* pada basis data adalah sebuah blok prosedur SQL yang berhubungan dengan tabel, view, skema, atau *database*. *Trigger* akan dieksekusi atau berjalan secara implisit pada saat sebuah kejadian tertentu terjadi [11].

Berdasarkan informasi yang dikemukakan oleh para ahli diatas, dapat disimpulkan bahwa *trigger* pada *database* adalah sebuah kode prosedural yang akan berjalan secara otomatis apabila terjadi eksekusi *Data Manipulation Language* (*DML*) pada tabel-tabel yang sudah ditetapkan untuk menjalankan suatu prosedur untuk menjaga integritas dan konsistensi data pada sebuah *database*.

2.8 *Relational Database Management System (RDBMS)*

Dilansir dari oracle.com, *Relational Database Management System* adalah sebuah fitur dari *database* yang menyimpan dan menyediakan akses ke titik data yang terkait antara satu dengan yang lainnya [12]

Menurut Margaret Rouse, *Relational Database Management System* adalah kumpulan dan kemampuan program yang memungkinkan untuk memperbarui, mengelola, dan berinteraksi dengan *relational database* [13].

Berdasarkan informasi yang dipaparkan diatas, dapat disimpulkan bahwa *Relational Database Management System* pada *database* adalah sebuah *environment database* yang menggunakan manajemen relasi untuk mengelola data agar data-data yang ada memiliki konsistensi dan integritas data yang baik lewat nilai unik disetiap tabel-tabel yang ada.

2.9 *HMAC SHA-256*

Dilansir dari technopedia.com, HMAC SHA-256 merupakan kode otentikasi pesan yang menggunakan teknik kunci kriptografi bersama dengan fungsi hash, dimana kunci kriptografi bersifat pribadi dan hanya boleh diketahui oleh server ataupun *client* tertentu [14].

Berdasarkan keterangan diatas, dapat disimpulkan bahwa HMAC SHA-256 merupakan bentuk penerapan keamanan yang dilakukan untuk menjaga keamanan dalam sebuah produk teknologi informasi dan kode bersifat pribadi (hanya boleh diketahui *user* yang memiliki *privilege* atau berkepentingan saja).

2.10 *SQL Injection*

Berikut ini merupakan definisi *SQL injection* menurut beberapa ahli, antara lain:

1. Menurut Mohammad Sadegh & Bahare Tajalli Pour, *SQL injection* adalah teknik serangan yang mengeksploitasi

kerentanan keamanan yang terjadi di lapisan *database* dan *service* yang ada pada aplikasi [15].

2. Menurut Satrio Gita Nugraha, Supeno Djanali, dan Baskoro Adi Pratomo, *SQL injection* adalah serangan yang merubah *query* normal pada aplikasi menjadi query berbahaya yang memungkinkan pengaksesan dan pemrosesan basis data secara tidak normal [16]

Berdasarkan informasi yang dikemukakan oleh para ahli diatas, dapat disimpulkan bahwa *SQL injection* adalah sebuah metode yang dilakukan oleh seseorang untuk masuk kedalam sistem *database* suatu organisasi atau perorangan dengan tujuan yang bervariasi. Biasanya *SQL injection* digunakan untuk tujuan yang tidak baik dan berpotensi merusak sistem basis data.

2.11 *Use Case Diagram*

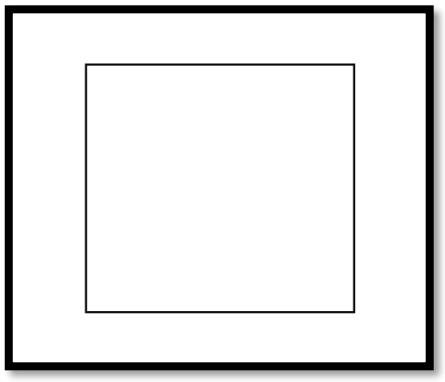
Menurut Michael Kharisma Hutauruk, *use case diagram* adalah diagram yang menggambarkan hubungan antara aktof dan sistem, biasanya digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem. [17]

Berdasarkan informasi yang dikemukakan diatas, dapat disimpulkan bahwa *use case diagram* yang masuk dalam bagian *Unified Modeling Language (UML)* merupakan sebuah struktur diagram yang bertujuan untuk memberikan gambaran mengenai relasi antara aktor dan juga fitur-fitur

utama pada sebuah sistem sehingga diagram dapat dengan mudah dipahami oleh konsumen.

Terdapat 3 komponen yang dimiliki oleh *use case diagram*, antara lain :

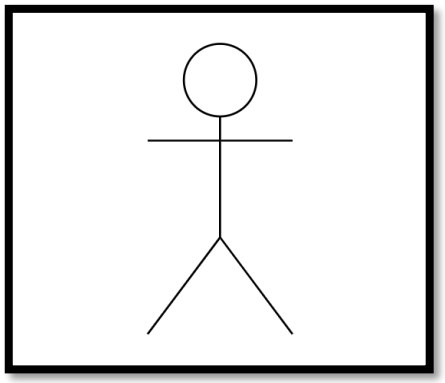
1. Sistem



Gambar 2. 1 Notasi Sistem pada *Use Case Diagram*

Visualisasi terkait batasan sistem yang dapat diakses oleh aktor-aktor yang ada dan fungsi-fungsi yang harus disediakan di dalam sistem.

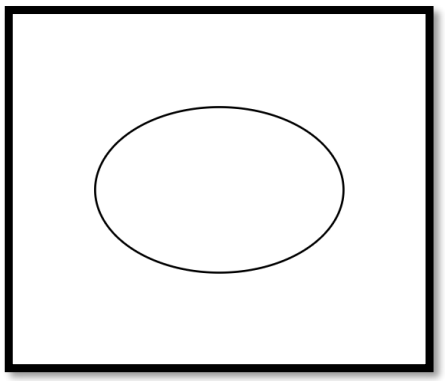
2. Aktor



Gambar 2. 2 Notasi Aktor pada *Use Case Diagram*

Visualisasi terkait segala hal yang berada di luar ruang lingkup sistem yang diperkirakan akan menggunakan sistem tersebut untuk melakukan sebuah pemrosesan atau transaksi.

3. *Use Case*

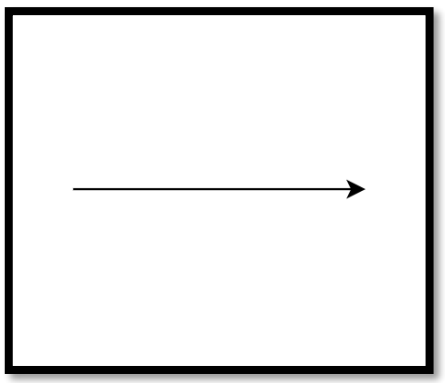


Gambar 2. 3 Notasi *Use Case* pada *Use Case Diagram*

Visualisasi fitur-fitur utama dalam sebuah sistem. Dengan objek ini, klien dan pengguna dapat memahami fungsi sistem yang akan, sedang, atau telah dibangun.

Terdapat 3 relasi yang dimiliki oleh use case diagram, antara lain :

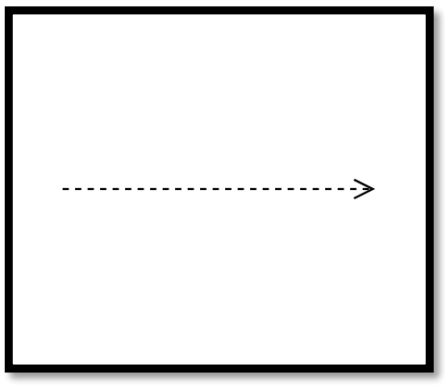
1. *Association*



Gambar 2. 4 Notasi Relasi Association pada Use Case Diagram

Interaksi yang dilakukan antara satu aktor dengan satu use case tertentu.

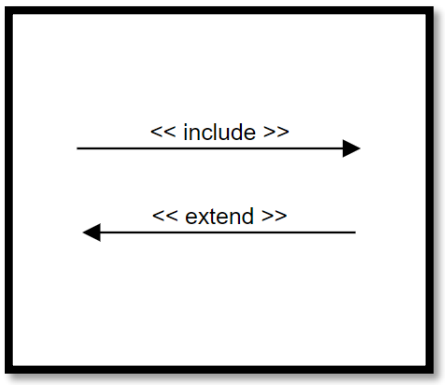
2. *Generalization*



Gambar 2. 5 Notasi *Generalization* pada *Use Case Diagram*

Interaksi yang dilakukan antara lebih dari satu aktor atau *use case* yang mana salah satunya menambahkan sifat dari yang lainnya.

3. *Dependency*



Gambar 2. 6 Notasi *Dependency* pada *Use Case Diagram*

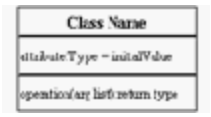
Pada relasi *dependency* terdapat 2 macam relasi, yaitu *include* dan *extend*. *Include* berfungsi untuk mengidentifikasi hubungan antara 2 *use case*, dimana salah satu *use case* akan memanggil *use case*

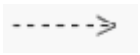
lainnya. *Extend* berfungsi apabila pemanggilan use case memerlukan kondisi tertentu.

2.12 *Class Diagram*

Dilansir dari visual-paradigm.com, *class diagram* adalah sebuah diagram struktur yang bersifat statis serta mendefinisikan struktur dari sistem dengan memperlihatkan kelas, atribut, metode atau operasi, dan juga relaasi diantara objek-objek yang ada pada sebuah sistem [18]. Notasi penulisan class diagram dapat dilihat pada tabel 2.1 :

***Tabel 2. 1 Notasi Penulisan* Class Diagram**

No	Gambar	Nama	Keterangan
1		*Association*	Menghubungkan antara objek satu dengan objek lainnya
2		*Aggregation*	Menghindari asosiasi yang lebih dari 2 objek
3		Composite	Hubungan yang lebih erat daripada asosiasi
4		*Class*	Kumpulan objek-objek yang memiliki atribut serta operasi yang sama
5		*Collabortation*	Deskripsi urutan aksi yang ditampilkan sistem
6		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek

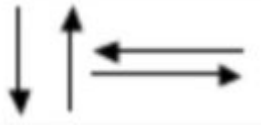
7		*Dependency*	Perubahan yang dilakukan operasi terkait terjadi pada elemen independent
8		*Boundary*	Digunakan untuk interaksi antara sistem dan juga aktor
9		*Control*	Digunakan untuk enkapsulasi control yang dihubungkan dengan use case yang spesifik
10		*Entity Class*	Digunakan untuk model yang ditujukan sebagai informasi

2.13 Activity Diagram

Menurut Muhammad Rizky, *activity diagram* merupakan sebuah visualisasi berbentuk rancangan struktur aliran aktifitas pada sebuah sistem yang akan dijalankan. Fungsinya yakni memperlihatkan urutan aktifitas pada sistem, dan membantu memahami proses secara keseluruhan [19]. Tabel 2.2 akan memperlihatkan komponen-komponen yang ada pada *activity diagram* adalah sebagai berikut :

Tabel 2. 2 Komponen Activity Diagram

No	Gambar	Nama	Keterangan
1		*Activity*	Objek yang memperlihatkan masing-masing kelas antar muka saling berinteraksi
2		*Action*	Objek yang memperlihatkan pemrosesan

			bahwa sistem melakukan suatu aksi
3		*Start*	Objek mengawali aktifitas
4		*End*	Objek mengakhiri aktifitas
5		*Decision*	Objek yang digunakan untuk menggambarkan suatu validasi atau pilihan (Y/N)
6		*Line Connector*	Objek yang digunakan untuk menghubungkan symbol satu dengan symbol lainnya

2.14 Metode Spiral

Menurut Junyanti, metode spiral adalah salah satu metode yang digunakan dalam pengembangan software. Metode spiral merupakan penggabungan dari model *prototyping* dan model *waterfall* [20].

Metode ini dikenal dengan sebutan “Spiral Boehm”. Pengembangan sistem dengan metode ini memiliki 5 tahapan, antara lain:

1. Tahap Liason

   Membangun komunikasi dengan setiap pihak yang terlibat pada proyek untuk menentukan hal apa yang menjadi tujuan akhir dari proyek terkait.

2. Tahap *Planning*

   Menentukan sumber-sumber informasi, *timeline* pengerjaan, dan I nformasi yang berhubungan dengan perencanaan proyek terkait.

3. Tahap *Risk Analysis*

   Mendefinisikan setiap resiko yang berpotensi terjadi (teknis ataupun manajemen).

4. Tahap *Engineering & Development*

   Melakukan pembuatan perangkat lunak berdasarkan rancangan yang telah dibuat sebelum tahap *engineering* dan *development* dilakukan.

5. Tahap *Evaluation*

   Melakukan evaluasi produk akhir atau jadi dari proyek terkait. Pada umunya, evaluasi ini diberikan oleh pengembang kepada *client* dari proyek.

2.15 Penelitian Terdahulu

Tabel 2. 3 Penelitian Terdahulu

Penulis	Judul	Nama Jurnal	Kesimpulan
Rjeib & Alsharqi, 2018	*Multi Mechanism Approach for Preventing SQL Injection Attacks in Stored Procedures*	*International Journal of Recent Trends in Engineering & Research*, Volume 04, No. 1, Januari 2018	Penelitian ini menghasilam sebuah pernyataan konfirmasi bahwa Stored procedure dapat mengamankan basis data (*database*) dari serangan *SQL injection*.
Jerzy Letkowski, 2015	*Doing Database Design with MySQL*	*Journal of Technology Research*, Volume 6, No. 1, Februari 2015	Penelitian ini menghasilkan sebuah pengetahuan cara melakukan perancangan pada database MySQL untuk menjaga konsistensi dan integritas data pada basis data terkait.
Yulianingsih, 2016	Menangkal Serangan *SQL Injection* dengan *Parameterized Query*	Jurnal Edukasi dan Penelitian Informatika (JEPIN) Volume 2, No. 1, Juni 2016	Penelitian ini menghasilam sebuah pembuktian secara nyata bahwa stored procedure terbukti dapat melindungi sistem basis data dari

Penulis	Judul	Nama Jurnal	Kesimpulan
			serangan SQL Injection dengan beberapa metode yang diuji
Rohmana, Mubarok, & Gunawan, 2019	Pengukuran Kinerja *Stored Procedure* pada *Database* Relasional	Jurnal Siliwangi, Volume 5, No.2, 2019	Penelitian ini menghasilam sebuah pembuktian secara nyata bahwa kinerja *stored procedure* lebih cepat dibandingkan dengan *query* biasa pada sistem basis data.
Pamungkas & Yuliansyah, 2017	Rancang Bangun Aplikasi Android POS (*Point of Sale*) Kafe untuk Kasir *Portable* dan *Bluetooth Printer*	Jurnal Sains & Teknologi, Volume 6, No.1, 2017	Penelitian ini menghasilkan sebuah aplikasi *point of sales* (*POS*) berbasis mobile yang dapat melakukan pencetakan struk menggunakan *printer bluetooth*
Rahmatulloh, Sulastri, dan Rizal Nugroho, 2018	Keamanan RESTful Web Service Menggunakan JSON Web Token (JWT) HMAC SHA-512	Jurnal Nasional Teknik Elektro dan Teknologi Informasi, Volume 7, No.2, 2018	Penelitian ini menghasilkan sebuah tata cara pengaplikasian HMAC SHA dalam aplikasi serta membandingkan kecepatan antara HMAC SHA-256 dengan SHA-512.

Berdasarkan penelitian-penelitian terdahulu yang ditunjukkan pada tabel diatas, maka topik penelitian mengenai rancang bangun aplikasi *point of sales* berbasis android dengan implementasi logika pemrograman & bisnis pada sistem basis data ini dapat diambil dan dilaksanakan.

Perbedaan antara penelitian ini dengan penelitian terdahulu yaitu penelitian ini menghasilkan produk akhir yang menggunakan konsep-konsep pada penelitian terdahulu, seperti penggunaan *stored procedure*, *function, trigger* pada basis data, hmach token, dan hal-hal lain yang telah disebutkan pada tabel penelitian terdahulu.