BAB II

LANDASAN TEORI

2.1 Sistem

"Sistem didefenisikan sebagai sekumpulan prosedur yang saling berkaitan dan saling terhubung untuk melakukan suatu tugas bersama-sama. Secara garis besar, sebuah sistem informasi terdiri atas tiga komponen utama yaitu software, hardware dan brainware. Ketiga komponen ini saling berkaitan satu sama lain." (Syahril & Nurlaila, 2020)

2.2 Perancangan

Menurut Rudi Setiyanto (2019), "Perancangan adalah sebuah proses untuk mendefinisikan sesuatu yang akan dikerjakan dengan menggunakan teknik yang bervariasi serta didalamnya melibatkan deskripsi mengenai arsitektur serta detail komponen dan juga keterbatasan yang akan dialami dalam proses pengerjaannya".

2.3 QR Code

Andy, Achyar dan Reynita (2017) berpendapat bahwa *QR code* adalah sebuah jenis kode matriks yang dapat menyimpan data secara horizontal dan vertikal sehingga dapat memuat data lebih banyak jika dibandingkan dengan *barcode*, *QR code* juga dapat dipindai dengan menggunakan aplikasi pada smartphone.

Dengan digunakannya *qr code* dalam transaksi di Minimarket Sastro Mart bisa menjadi lebih praktis dan data yang disimpan lebih aman serta sesuai dengan riwayat transaksi *customer*.

2.4 Minimarket

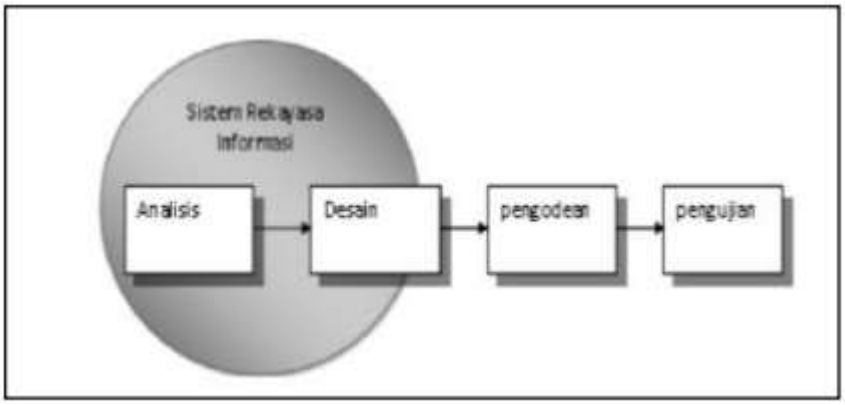
Menurut Soraya Rahma Avelina (2018) memaparkan bahwa: “Pengetian minimarket secara kata merupakan gabungan dari kata “mini” yang berarti kecil dan “market” yang berarti pasar. Jadi minimarket adalah sebuah pasar kecil yang menjual barang-barang bervariatif dan lengkap seperti di dalam pasar”.

2.5 Metode Pengembangan Sistem

Dalam pengembangan sistem ini, metode pengembangan yang digunakan adalah metode system development life cycle dengan Model Waterfall**.**

2.5.1 Model *Waterfall*

Menurut (Muhammad Susilo, 2018:100) Model waterfall adalah yang paling banyak digunakan untuk tahap pengembangan. Model waterfall ini juga dikenal dengan nama model tradisional atau model klasik.Model air terjun (*waterfall*) sering juga disebut model sekuensial linier (squential linear) atau alur hidup klasik (*Classic cycle*)”. Model air terjun ini menyediakan pendekatan alur hidup perangkat lunak secara sekuensial terurut dimulai dari analisis, desain, pengkodean, pengujian dan tahap pendukung (*support*).



Gambar 2.1 Model *Waterfall* (Muhammad Susilo, 2018)

Adapun penjelesan urutan tahapan-tahapan yang dimiliki oleh waterfall adalah sebagai berikut :

1. Analisa Kebutuhan Sistem

   Berdasarkan kebutuhan pengguna dalam sistem yang diperoleh melalui aktivitas yang berlangsung, penulis dapat menganalisa perlunya dirancang sebuah aplikasi yang mempunyai fungsi tertentu yang mampu memenuhi standar proses dalam sistem. Adapun dalam model ini merupakan dalam tahapan analisa kebutuhan dari sistem yang terdiri dari menu-menu yangdiperlukan dalam sistem yang akan dirancang.

2. Desain

   Tahap desain dalam penelitian ini meliputi rancangan sistem menggunakan diagram *use case*, *activity diagram* dan rancangan database dengan menggunakan diagram *entity relationship* diagram dan *logical record structure*. Adapun *use case* diagram merupakan sebuah diagram yang dapatdigunakan untuk menggambarkan hubungan sistem dan *user* dengan kasus yang disesuaikan sesuai langkah-langkah yang sudah ditentukan.

3. Pengkodean

   Setelah tahap desain dilakukan, desain yang telah dibuat maka desain tersebut harus ditranslansikan kedalam sebuah program perangkat lunak, yang hasilnya menjadi sebuah aplikasi sistem informasi sesuai dengan desain yang telah dibuat pada tahap desain.

4. Pengujian

   Dalam tahap pengujian ini merupakan tahap yang terfokus pada pengujian aplikasi sistem informasi dari segi logic dan fungsional serta memastikan aplikasi sistem informasi yang dirancang bahwa semua bagian sudah diuji. Tahap ini dilakukan agar dapat meminimalisir *error* serta keluaran yang hasilkan dapat dipastikan sesuai yang di rancang.

5. *Support* atau *Maintenance*

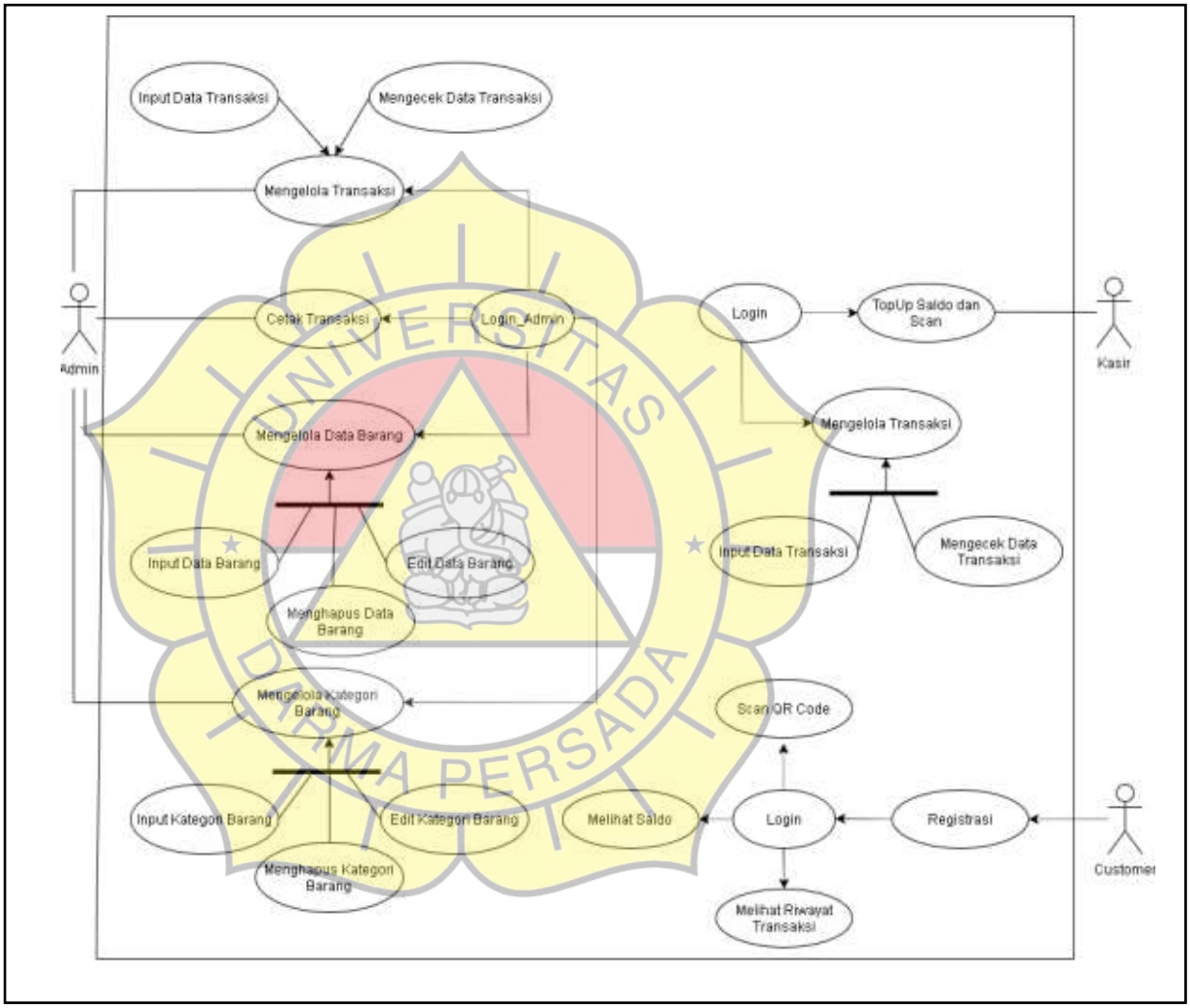
   Tahap ini merupakan tahap dalam melakukan pengembangan atau adanya perubahan dalam sistem yang dibuat terkait dengan ditemukan adanya sedikit error yang baru ditemukan atau tidak ditemukan sebelumnya dan adanya penambahan fitur dalam sistem yang belum ada pada sistem sebelumnya, faktor tersebut pula dapat dipengaruhi jika terdapat perubahan perangkat, sistem operasi ataupun perubahan dalam alur proses bisnis perusahaan.

2.5.1.1 Analisa Kebutuhan

Pada tahap ini, merupakan proses analisa kebutuhan sistem. Pengembang mengumpulkan data-data sebagai bahan pengembangan sistem. Pengumpulan data dapat dilakukan dengan teknik wawancara, teknik observasi, dan teknik kuisioner. Dalam Tahapan ini akan menghasilkan dokumen user requirement atau bisa dikatakan sebagai data yang berhubungan dengan keinginan user dalam pembuatan sistem.

2.5.1.2 Design

Tahapan kedua dari model waterfall adalah desain dimana pada tahapan ini bertujuan membuat desain dari hasil analisis yang dilakukan pada tahapan pertama. Informasi, model dan spesifikasi yang diubah menjadi sebuah desain sistem yang nantinya akan dikodekan.



Gambar 2.*2 Use Case* Diagram Sistem

Gambar *use case* diagram diatas merupakan gambaran dari sistem ini, dimana terdapat 3 aktor yang berperan, yakni Admin, Kasir dan *Customer*.

2.5.1.3 Pengkodean

Desain harus ditranslasikan ke dalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai dengan desain yang telah dibuat pada tahap desain

2.5.1.4 Pengujian

Pengujian bertujuan menemukan kesalahan pada sistem dan mencari tahu kesesuaian sistem yang dibuat dengan kebutuhan pengguna. Pengujian dilakukan pada aspek fungsionalitas. Pengujian fungsionalitas berfungsi menguji kelayakan sistem informasi pembayaran dengan *qr code* ini. Pengujian dilakukan kepada admin /pemilik toko, kasir, dan *customer*.

2.5.1.5 Support dan *Maintenance*

Tidak menutup kemungkinan sebuah piranti lunak akan mengalami perubahan ketika sudah dikirimkan ke pengguna. Perubahan dapat terjadi karena adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian, atau dapat juga terjadi piranti lunak atau software harus beradaptasi dengan lingkungan baru.

2.5.1.5.1 Website

Qitvirul, Achmad dan Mohammad (2020) berpendapat bahwasanya *website* merupakan sekumpulan halaman yang menampilkan data- data berupa teks, gambar, audio maupun video baik yang bersifat statis maupun dinamis yang terkait satu sama lain yang dihubungkan oleh jaringan-jaringan halaman.

2.5.1.5.2 Android

Android adalah sistem operasi yang berbasis *Linux* untuk telepon seluler seperti telepon pintar dan komputer tablet. *Android* menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. Awalnya, Google Inc. membeli *Android* Inc., pendatang baru yang membuat peranti lunak untuk ponsel. Kemudian untuk mengembangkan *Android*, dibentuklah *Open Handset Alliance*, konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, TMobile, dan Nvidia. Pada saat perilisan perdana *Android*, 5 November 2007, *Android* bersama *Open Handset Alliance* menyatakan mendukung pengembangan standar terbuka pada perangkat seluler. Di lain pihak, Google merilis kode– kode *Android* di bawah lisensi Apache, sebuah lisensi perangkat lunak dan standar terbuka perangkat seluler.(Harni & Nicky, 2016).

2.1.5.1.3 *HTML*

Menurut Jubilee Enterprise (Penerbit PT Elex Media Komputindo, 2016) dalam Buku *"Pengenalan HTML dan CSS". HTML (Hyper Text Markup Language)* adalah sebuah text berbentuk link dan mungkin juga foto atau gambar yang saat di klik, akan membawa si pengakses internet dari satu dokumen ke dokumen lainnya.

2.1.5.1.4 *CSS*

Menurut Fahrul Minokaura et.al.(2019) menjelaskan bahwa, "*Cascading Style Sheets* atau lebih dikenal dengan *CSS* merupakan kumpulan kode yang terkandung dalam *HTML* dan bertujuanuntuk mengatur gaya atau tampilan dari sebuah halaman *website*".

2.1.5.1.5 *PHP*

PHP adalah bahasa yang dirancang secara khusus untuk penggunaan pada *Web*. *PHP* adalah tool untuk pembuatan halaman web dinamis. Pada awalnya *PHP* merupakan kependekan dari *Personal Home Page* (Situs Personal). *PHP* pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada waktu itu *PHP* masih bernama FI (*Form Interpreted*), yang wujudnya berupa sekumpulan script yang digunakan untuk mengolah data form dari web.Saat ini *PHP* adalah singkatan dari *PHP:Hypertext Preprocessor*, sebuah kepanjangan rekursif, yakni permainan kata dimana kepanjangannya terdiri dari singkatan itu sendiri: *PHP: Hypertext Preprocessor*.(Ahmad Lutfi, 2017).

2.1.5.1.6 *MySQL*

Menurut Amik (2019), *MySQL* adalah sebuah *database manajemen system* (DBMS) popular yang memiliki fungsi sebagai relational database manajemen *system* (RDBMS). Selain itu *MySQL* software merupakan suatu aplikasi yang

sifatnya open source serta server basis data *MySQL* memiliki kinerja sangat cepat, reliable, dan mudah untuk digunakan serta bekerja dengan arsitektur client 3 *server* atau *embedded systems*. *MySQL* digunakan untuk membuat dan mengolah database dengan isinya. *MySQL* juga bermanfaat untuk menambahkan, mengubah dan menghapus data yang berada dalam database. Data-data yang dikelola dalam database diletakkan dalam table yang terpisah dan memanipulasi data akan menjadi jauh lebih cepat.

2.1.5.1.7 *Javascript*

Tewuh, Brave dan Alicia (2019) mengemukakan bahwa *Javascript* adalah bahasa pemrograman web yang bersifat *Client Side Programming Language*. *Client Side Programming Language* adalah tipe Bahasa perograman yang pemrosesannya dilakukan oleh *Client*.*JavaScript* pada awal perkembangannya berfungsi untuk membuat interaksi natar *user* dengan situs web menjadi lebih cepat tanpa harus menunggu pemrosesan di web server. Berbagai animasi untuk mempercantik halaman web, fitur chatting, efek-efek modern, games, semuanya bisa dibuat menggunakan *JavaScript*.

2.1.5.1.8 *JQUERY*

Menurut Ahmad Sahi (2020), *Jquery* merupakan suatu *framework* (library) *Javascript* yang menekankan bagaimana

interaksi antara *Javascrip*t dan *HTML*. *JQuery* pertama kali dirilis pada tahun 2006 oleh John Resig. Fitur utama dari *JQuery* diantaranya :

a) Dapat mengakses elemen dalam dokumen Javascript khusus, untuk mengakses suatu bagian tertentu dari halaman, harus mengikuti aturan *Document Object Model* dan pengaksesan harus secara spesifik menyesuaikan dengan struktur *HTML*.
b) Mengubah tampilan halaman website CSS (*Cascading Style Sheet*) menawarkan metode yang cukup handal dalam mengatur dan mempercantik halaman web.
c) Merespon interaksi *user Javascript* sendiri memiliki beberapa event-handling seperti onclick untuk menangani event saat terjadi click.

2.5.1.6 *Unified Modeling language* (UML)

UML atau Unified Modeling Language merupakan sebuah metode pemodelan secara visual yang berfungsi sebagai sarana perancangan sistem berorientasi objek, tujuan dari pemodelan ini adalah untuk memodelkan sistem perangkat lunak dari segi pembangunan, produktifitas, kualitas, pengurangan biaya dan juga waktu.

Menurut Harco Leslie (2017) *UML* adalah sebuah bahasa standar yang digunakan untuk memodelkan proses bisnis, memodelkan tahap-

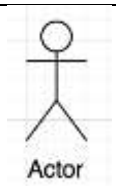
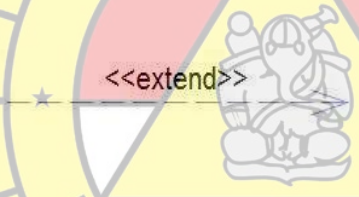
tahap pengembangan sebuah sistem yaitu tahap analisa, desain dan penerapan aplikasinya.

2.5.1.7 Use Case Diagram

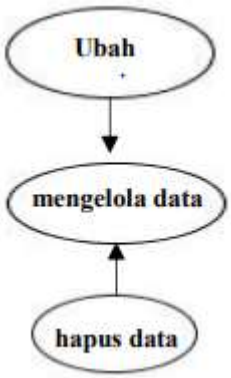
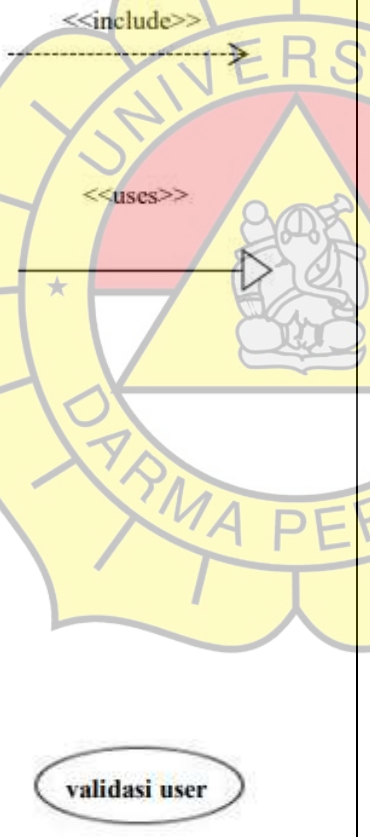
Menurut Julianto & Setiawan (2019), *Diagram use case* merupakan pemodelan untuk kelakuan sistem informasi yang akan dibangun. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibangun. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada pada sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Berikut ini adalah simbol-simbol diagram *use case*, seperti yang terlihat pada tabel 1 dibawah ini :

Tabel 2.1 Simbol *Use Case* (Julianto & Setiawan, 2019)

Simbol	**Deskripsi**
Use Case 	*Funsionalitas* yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja diawal di awal frase nama *use case*
Aktor/Actor	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu

	merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
Asosiasi/Association _____	Komunikasi antara *actor* dan *use case* yang berpatisipasi pada *use case* atau *use case* memiliki interaksi dengan *actor*.
Eksestensi/Extend 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu; mirip dengan prinsip inheritance pada pemrograman berorientasi objek; biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan misal Arah panah mengarah pada *use case* yang ditambakan; biasanya *use case* yang menjadi extend-nya merupakan jenis yang sama dengan *use case* yang menjadi induknya.

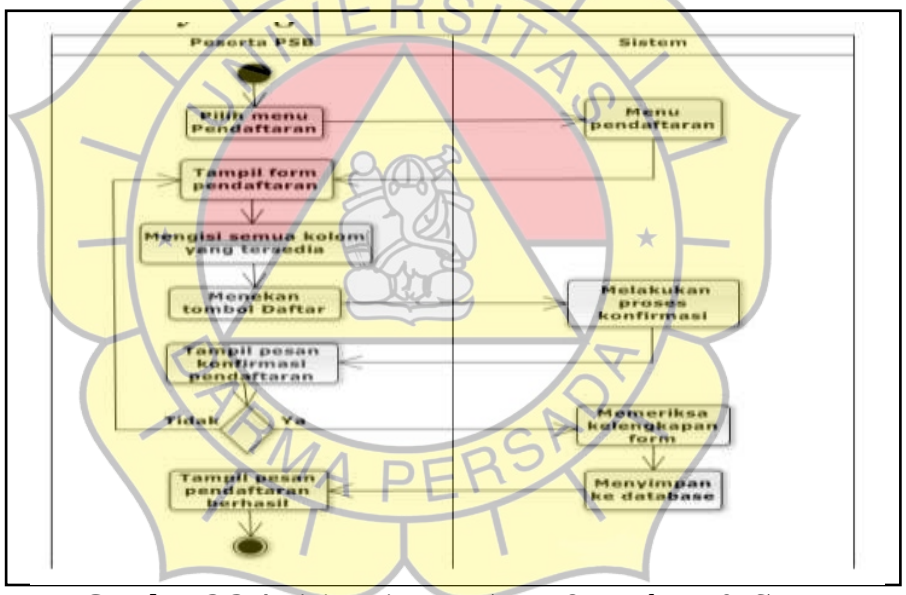
Generelasi 	Hubungan generalisasi dan spesialisasi (umum – khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya, misalnya: arah panah mengarah pada *use case* yang menjadi generalisasinya (umum).

	
Menggunakan/ *include* / *uses* 	Relasi *use case* tambahan ke sebuah *use case* di mana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini ada dua sudut pandang yang cukup besar mengenai include di *use case*: 1. Include berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan, misal pada kasus berikut: 2. Include berarti *use case* yang tambahan akan selalu melakukan pengecekan apakah *use case* yang ditambahkan telah dijalankan sebelum *use case* tambahan dijalankan, misal pada kasus berikut: Kedua interpretasi diatas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan.

2.5.1.8 Activity Diagram

Menurut Julianto & Setiawan (2019), Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Penekanan pada diagram aktivitas adalah menggambarkan aktivitas sistem atau aktivitas yang dapat dilakukan oleh sistem, bukan apa yang dilakukan *aktor*.

Activity diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem dan *user* (Regi & Hanhan, 2016)

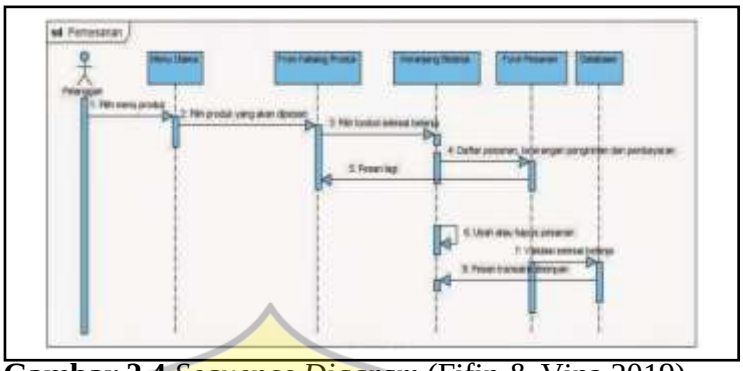


Gambar 2.3 *Activity Diagram* (Regi & Hanhan 2016)

2.5.1.9 Sequence Diagram

Menurut Julianto & Setiawan (2019), Diagram sekuen "menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan massage yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambar diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta

metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu". Membuat diagram sekuen juga dibutuhkan untuk melihat skenario yang ada pada *use case*.



Gambar 2.4 *Sequence Diagram* (Fifin & Vina 2019)

2.5.1.10 Data Mining

Agus, Nunu dan Wanti (2018) berpendapat bahwa *data mining* adalah suatu proses menemukan hubungan yang berarti, pola, dan kecenderungan dengan memeriksa dalam sekumpulan besar data yang tersimpan dalam pemyimpanan dengan menggunakan teknik pengenalan pola seperti teknik statistik dan matematika.

Ade, Lince dan Taronishoki (2018) juga mengemukakan bahwa *data mining* merupakan pengetahuan yang tersembunyi didalam database yang di proses untuk menemukan pola dan teknik statistik matematika, kecerdasan buatan, dan *machine learning* untuk mengekstraksi dan mengidentifikasi informasi pengetahuan dari database tersebut.

2.6 Algiritma yang Digunakan

2.6.1 Algoritma *FP-Growth*

Algoritma *Fp –Growth* merupakan salah satu algoritma yang dapat digunakan untuk menentukan himpunan data yang sering muncul dalam sebuah *itemset*.

Russy & Dito, (2019) Algoritma *FP-Growth* merupakan pengembangan dari algoritma Apriori. Algoritma *Frequent Pattern Growth* (*FP-Growth*) adalah salah satu alternatif algoritma yang dapat digunakan untuk menentukan himpunan data yang paling sering muncul (*frequent itemset*) dalam sebuah kumpulan data. Pada algoritma *FP-Growth* menggunakan konsep pembangunan *tree*, yang biasa disebut *FP-Tree*, dalam pencarian *frequent itemset* bukan menggunakan *generate candidate* seperti yang dilakukan agar dapat memberikan hasil yang maksimal, tahapan-tahapan tersebut yaitu :

1. Tahap pembangkitan *conditional pattern base*.
2. Tahap pembangkitan *conditional FP-* pada algoritma *Apriori.* Dengan menggunakan konsep tersebut, algoritma *FP-Growth* menjadi lebih cepat daripada algoritma *Apriori*. Algoritma *FP-Growth* memiliki tahapan-tahapan yang harus dilewati *Tree.*
3. Tahap pencarian frequent *itemset*. *Association rule* merupakan sutu proses pada data mining untuk menentukan semua aturan asosiatif yang memenuhi syarat *minimum* untuk *support (minsup) dan confidance (minconf)* pada sebuah database. Kedua syarat tersebut akan digunakan untuk *interesting*

association rules dengan dibandingkaan dengan batasan yang telah ditentukan, yaitu *minsup* dan *minconf*.

Association Rule Mining adalah suatu prosedur untuk mencari hubungan antar item dalam suatu *dataset*. Dimulai dengan mencari *frequent itemset*, yaitu kombinasi yang paling sering terjadi dalam suatu itemset dan harus memenuhi *minsup*. Dalam tahap ini akan dilakukan pencarian kombinasi item yang memenuhi syarat *minimum* dari nilai *support* dalam database.Untuk mendapatkan nilai *support* dari suatu item A dapat diperoleh dengan rumus berikut

$$Support\ A = \frac{Jumlah\ Transaksi\ Mengandung\ Item\ A}{Total\ Transaksi}$$

Gambar 2.5 Rumus untuk mencari nilai *Support A*

$$Support\ (A,B) = P(A \cap B) = \frac{Jumlah\ Transaksi\ Mengandung\ A\ dan\ B}{Total\ Transaksi}$$

Gambar 2.6 Rumus untuk mendapatkan nilai *support* dari dua item

$$Confidence\ (A \rightarrow B) = P(AB) = \frac{Jumlah\ Transaksi\ Mengandung\ A\ dan\ B}{Jumlah\ Transaksi\ yang\ Mengandung\ A}$$

Gambar 2.7 Rumus untuk mencari syarat umum *minimum confidence*

2.6.2 Algoritma *Hash Based*

Triaji, Iin dan Henki (2020), Algoritma *Hash Based* merupakan algoritma yang termasuk dalam golongan *frequent itemset* mining yang bertujuan untuk mencari *itemset* yang memenuhi *minimum support*. Dalam algoritma *Hash Based*

penggunaan teknik *hashing* dalam pembangkitan *itemset* selanjutnya dengan cara menyaring *itemset* yang tidak penting.

Kemudian kandidat k-*itemset* dari *support count* dihitung dengan menelusuri basis data, algoritma *Hash Based* akan mengumpulkan informasi tentang (k+1)-*itemset* dengan cara seluruh kemungkinan (k+1)-*itemset* di *hash* kedalam tabel *hash* dengan mengunakan sebuah bilangan prima untuk operasi *modular*. Didalam setiap *bucket* tersebut pada *hash table* berisi berapa kali *itemset* telah di *hash.* Berdasarkan *hash table* tersebut akan dibuat *node* dimana *node* bernilai 1 jika angka pada *bucket* yang bersanguktan lebih besar atau sama dengan *minimumsupport*. Pembagkitan kandidat yang telah menghitung Ck=Lk-1*Lk-1, pada setiap k-*itemset* diperiksa apakah *itemset* tersebut dapat di *hash* kedalam *bucke*t yang memiliki node yang sama dengan yang pertama. Namun *itemse*t tidak dipergunaan bila tidak ada yang sama dengan yang pertama. Dengan menggunakan tabel *hash* dapat mengurangi jumlah kandidat k-*itemset* kemudian juga dapat mengurangi nilai komputasi dari pembangkit *itemset* pada setiap iterasinya.

Algoritma hash juga dapat membantu untuk mendeteksi apakah ada yang diubah atau data yang berubah karena kesalahan jaringan, maka dari itu untuk menggunakan *smart contract* transaksi pembuatan khusus dijalankan, selama prosedur ini , kontrak diberikan alamat unik dalam bentuk identifikasi. Setelah berhasil dibuat, kontrak pintar terdiri dari alamat kontrak, saldo contract, kode yang dapat dieksekusi yang ditentukan sebelumnya, dan status. Pihak yang berbeda kemudian dapat berinteraksi dengan kontrak tertentu dengan

mengirimkan transaksi yang melibatkan kontrak ke alamat contract yang diketahui

Pada jurnal "*A Hash Based Frequent Itemset Mining Using Rehashing*", menyatakan bahwa mengusulkan algoritma aturan asosiasi mining baru yang di sebut RBFI dimana teknologi *hashing* yang ada digunakan untuk menyumpan basis data kedalam format data *vertical*. Cara kerjanya dimana semua basis data akan direpresentasikan kedalam basis data vertikal, untuk menemukan dan menghasilkan *frequent itemset* baru dimana:

$$n = 2 * m + 1$$

Gambar 2.8 Rumus *Hash Based* (Triaji, 2020)

Keterangan :

m : jenis makanan pada seluruh pesanan

n : banyak alamat

Setelah didapatkan nilai n, perhitungan akan dilanjutkan, namun apa bila terjadi tabrakan (*collision*) harus segera dilakukan pengecekan untuk mencari alamat *bucket* yang masih kosong. Kemudian melakukan teknik r*ehashing* ulang dengan banyak alamat 2 kali dari alamat sebelumnya. Untuk menyelesaikan masalah tersebut dipergunakan rumus yang berbeda yaitu:

$$j = 2 * m + 1$$

Gambar 2.9 Rumus *Hash Based* 1 (Triaji, 2020)

Setelah alamat berubah karena terdapatnya collision maka rumus *hash* menjadi:

$$h(k) = \big((Order\ Of\ Item\ X) * 10 + Order\ Of\ Item\ Y\big) mod\ j$$

Gambar 2.10 Rumus *Hash Based 2* (Triaji, 2020)