BAB II

LANDASAN TEORI

2.1 Tinjauan Pustaka

Adapun perbedaan penelitian dengan penelitian terdahulu sebagai berikut :

1. Andre and Prastowo (2020) meneliti tentang Sistem Informasi *Order* Jasa Pariwisata (Study Kasus : Musa Tour Lampung). Metode pengembangan sistem yang digunakan adalah metode *waterfall*, dan perancangan sistem menggunakan perancangan sistem UML. Pengamatan, wawancara, dokumentasi dalam pengolahan data pemesanan pariwisata juga digunakan untuk menambah kedekatan keakuratan data. Hasil yang dicapai dalam tulisan ini adalah sebuah sistem jasa tour pariwisata untuk meningkatkan jangkauan pemesanan perusahaan, maka Musa Tour Lampung ingin membangun sarana pemesanan berbasis *web*, sistem ini di fokuskan untuk mengelola data pemesanan sehingga dapat menghasilkan laporan yang dibutuhkan oleh perusahaan.
2. Adrian and Pramono (2017) meneliti tentang Rancang Bangun Sistem Informasi Penjualan Barang Pada Toko Distro Black Outlet Berbasis *Web*, hasil penelitian yaitu *E-Marketing*, untuk menyediakan informasi produk. Penelitian ini bertujuan untuk membuat system pemasaran dengan pendekatan *user experience Dsaign*. Penelitian ini menggunakan perangkat CMS *open cart.*
3. Fitriyana and Sucipto (2020) meneliti tentang Sistem Informasi Penjualan Oleh Sales Marketing Pada PT Erlangga Mahameru. Hasil dari penelitian ini adalah sistem informasi yang dapat mempermudah bagian marketing untuk

mempromosikan buku. Selain itu konsumen juga dapat melakukan transaksi secara *online*. Sistem informasi ini diuji menggunakan *black box* serta dilakukan pengukuran pengujian menggunakan quisioner. Hasil pengujian adalah sebesar 89,58%. Dengan demikian sistem yang dikembangkan ini layak untuk di implementasikan.

4. Alfiah and Damayanti (2020) meneltii tentang Aplikasi *E-Marketplace* Penjualan Hasil Panen Ikan Lele (Studi Kasus: Kabupaten Pringsewu Kecamatan Pagelaran). Kurangnya informasi mengenai harga ikan lele membuat pelanggan atau calon pembeli merasa rugi jika ternyata ada beberapa petani yang memberikan harga yang murah. Oleh sebab itu penjualan ikan lele dapat di tunjang dengan memanfaatkan teknologi informasi sebagai pendukung dalam proses penjualan dan pemasaran ikan lele. Hasil yang dicapai adalah Aplikasi *E-Marketplace* penjualan hasil panen ikan lele ini dapat memberikan kemudahan bagi penjual dan pembeli dalam memasarkan ikan lele serta mendapatkan informasi mengenai harga dan stok ikan lele.
5. Juniansyah, Susanto and Wahyudi (2020) meneliti tentang Pembuatan *E-Commerce* Pemesanan Jasa *Event Organizer* Untuk Zero Seven Entertaiment. *Website* yang dibuat adalah *event organizer*, dimana *event organizer* tersebut terdiri dari tiga *event* yaitu *event food festival*, modifikasi mobil, dan modifikasi motor menggunakan metode *extreme programming* dimana tahapannya berupa tahapan *planning, Dsaign, coding*, dan *testing, website* ini dilakukan secara *online* cara melakukan pemesanan *event* melalui *website* nya langsung supaya dapat diakses dimanapun dan kapanpun agar dapat memudahkan para pengguna *website* tersebut, supaya acara tersebut berjalan lancar

Berdasarkan penelitian terdahulu kaitan dan perbedaan dengan peneliti yaitu:

1. Memasarkan cetak kartu undangan
2. Metode pengembangan sistem yang digunakan dalam penelitian ini adalah metode *extreme programming*
3. Perancangan sistem menggunakan aplikasi *Jqwery Mobile* dan *MySQL* sebagai *database management system (DBMS).*

2.2 Sistem

Sistem adalah kumpulan atau himpunan dari unsur atau variable-variabel yang saling terkait, saling berinteraksi, dan saling tergantung satu sama lain untuk mencapai tujuan (Tohari, 2017). Sistem juga dapat didefinisikan sebagai kumpulan dari elemen-elemen berupa data, jaringan kerja dari prosedur-prosedur yang saling berhubungan, sumber daya manusia, teknologi baik *hadware* dan *software* yang saling berinteraksi sebagao kesatuan untuk mencapai tujuan atau sasaran tertentu yang sama (Maniah and Haminidin, 2017). Dari beberapa kutipan di atas maka penulis dapat menyimpulkan bahwa sistem informasi adalah sistem di dalam suatu instansi atau organisasi perusahaan yang mempertemukan kebutuhan pengolahan transaksi harian dan memberikan laporan-laporan atau informasi yang dibutuhkan.

2.3 Informasi

Informasi merupakan kumpulan data yang diolah menjadi bentuk yang lebih berguna dan lebih berarti bagi yang menerima. Tanpa suatu informasi, suatu sistem tidak akan berjalan dengan lancar dan akhirnya bisa mati. Suatu organisasi tanpa adanya suatu informasi maka organisasi tersebut tidak bisa berjalan dan tidak bisa beroperasi (Kristanto, 2018).

2.4 Sistem Informasi

Menurut Kristanto (2018) Sistem informasi merupakan suatu sistem yang dibuat oleh manusia yang terdiri dari komponen-komponen dalam organisasi untuk mencapai suatu tujuan yaitu menyajikan informasi.

2.5 Pemesanan

Pemesanan yaitu pemesanan fasilitas yang diantaranya akomodasi, meal, seat pada pertunjukan, pesawat terbang, kereta api, bus, hiburan, night club, *discoutegue* dan sebagainya merupakan proses, perbuatan, cara memesan (tempat,barang,dsb) kepada orang lain (McLoad, 2014)

Berdasarkan pengertian para ahli diatas dapat disimpulkan bahwa pemesanan adalah suatu aktifitas yang dilakukan oleh konsumen sebelum membeli atau melakukan perbaikan untuk mewujudkan kepuasan konsumen maka perusahaan harus mempunyai sebuah sistem pemesanan yang baik.

Pemesanan dibagi menjadi dua jenis yaitu pemesanan *online* dan pemesanan *offline* :

1) Pemesanan *Online.*

Kemajuan teknologi saat ini mengakibatkan sistem pemesanan juga mengalami perkembanga kearah sistem pemesanan online. Pemesanan online bisa diakses oleh siapapun dan dimanapun mereka berada yang memiliki akses internet.

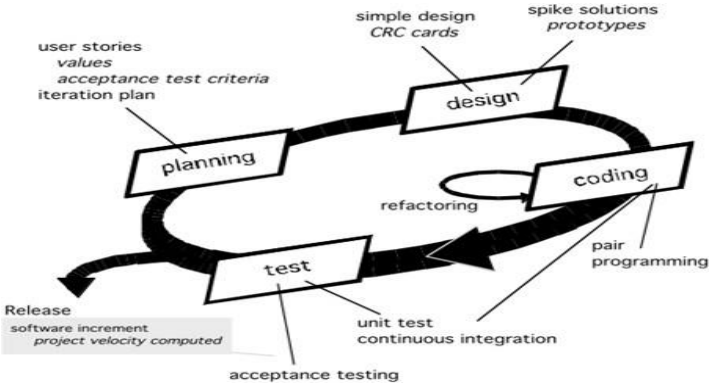
2) Pemesanan *Offline*

Sistem pemesanan yang menggunakan pengiriman pemesanan langsung ke tempat dengan media pemesanan seperti telepon, *fax*, *e-mail,* dan *walk in.*

2.6 Pengembangan Sistem *Extreme Programming*

Menurut Supriyatna (2018) *Extreme Programming (XP)* merupakan sebuah proses rekayasa perangkat lunak yang cenderung menggunakan pendekatan berorientasi objek dan sasaran dari metode ini adalah tim yang dibentuk dalam skala kecil sampai medium serta metode ini juga sesuai jika tim dihadapkan dengan *requirement* yang tidak jelas maupun terjadi perubahan–perubahan *requirement* yang sangat cepat.

Sedangkan menurut Lubis (2016) *Extreme Programming (XP)* dikenal dengan metode atau "*technical how to*" bagaimana suatu tim teknis mengembangkan perangkat lunak secara efisien melalui berbagai prinsip dan teknik praktis pengembangan perangkat lunak. XP menjadi dasar bagaimana tim bekerja sehari- hari. Tahapan *Extreme Programming* dapat dilihat pada **Gambar 2.1**



Gambar 2. 1 Model *Extreme Programming (XP)*
Sumber : Lubis (2016)

Berikut ini adalah penjelasan tahapan *Extreme Programming* yaitu :

1. *Planning* (Perencanaan)

Kegiatan Perencanaan (disebut juga *planning game*) biasanya dimulai dengan mendengarkan suatu kegiatan yang bertujuan mengumpulkan kebutuhan-kebutuhan untuk memahami konteks bisnis dan perlunya keluaran-keluaran

(*output*), fungsi utama, dan *fungsionalitas.* Pada perencanaan terdapat *user stories values* yaitu story dengan value tertinggi akan dipindahkan dari jadwal dan diimplementasikan pertama, *acceptance test criteria interation plan* melakukan perhitungan kecepatan project selama development, customer dapat menambah story, merubah value, membagi story atau menghapusnya.

2. *Dsaign* (Perancangan)

Perancangan yang simple, menarik, dan sederhana selalu memberikan hasil yang lebih disukai daripada gambaran-gambaran yang lebih kompleks. Perancangan XP memberikan panduan implementasi untuk suatu cerita ketika ditulis, tidak kurang, tidak lebih. Terdapat *simple Dsaign CRC Casrds* untuk mengenali dan mengatur *object oriented class* sesuai dengan *software increment* dan *spike solutions prototypes* melakukan spesifikasi solusi dari *object oriented class.*

3. *Coding* (Pengkodean)

Pengkodean ini dilanjutkan setelah cerita yang telah dikembangkan dan rancangan yang telah dilakukan oleh tim perangkat lunak. Pengkodean ini tidak langsung mengarah ke kode-kode program. Tim akan mengembangkan serangkaian unit pengujian lalu beralih ke pengkodean. Pada tahapan *pair programming* melakukan kerja sama untuk membuat code dari satu story. Dan *refactoring* adalah proses restrukturisasi kode program komputer yang ada tanpa mengubah perilaku eksternalnya.

4. *Pengujian* (Pengujian)

Unit pengujian yang harus dibuat dan kemudian dijalankan menggunakan kerangka kerja yang memungkinkan mereka untuk diotomatisasi sehingga dapat

dijialankan dengan mudah dan dapat dijalankan berulang kali. Pada tahapan pengujian yaitu *unit test continuos integration* yaitu tahapan pengujian code yang diintegrasikan dengan kerja lainnya dengan pengujian yang dilakukan oleh customer dan focus pada keseluruhan dan fungsional sistem, dan *acceptance testing* yaitu pengujian yang dilakukan *customer* stories yang akan diimplementasikan sebagai bagian dari *software realease.* Selanjutnya terdapat tahapan *software increment project velocity computed* yaitu tahapan yang telah diimplementasikan dari *software realease* yang nantinya akan diterapkan dalam suatu sistem.

2.7 UML (*Unified Modeling Language*)

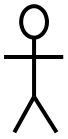
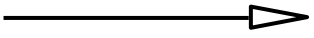
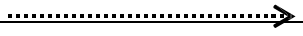
Menurut (Rosa and Shalahudin, 2018) *Unified Modeling Language (UML)* adalah bahasa standar untuk menulis perangkat lunak dalam bentuk gambar.*UML* dapat digunakan untuk memvisualisasikan, menentukan, membangun, dan mendokumentasikan sebuah sistem perangkat lunak. Beberapa jenis diagram *UML* antara lain sebagai berikut:

1. *Use Case Diagram*

Menurut (Rosa and Shalahudin, 2018) *use case* diagram membantu anda menentukan fungsi dan fitur dari perangkat lunak. Dalam diagram ini, gambar yang menyerupai boneka kayu mewakili aktor yang berhubungan dengan kategori dari pengguna. Di dalam diagram *use case.* Para aktor terhubung oleh garis ke *use case* yang mereka kerjakan.

Use case mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. simbol-simbol yang ada pada diagram *use case* dapat dilihat pada gambar 2.2 di bawah ini:

Tabel 2. 1 Simbol Diagram *Use Case*

Simbol	Deskripsi
Use Case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case*
Aktor/*actor* 	Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
Asosiasi/*association* 	Komunikasi antara aktor dan *use case* yang berpatisipasi pada *use case* atau *use case* memiliki interaksi dengan actor
Ekstensi/*extend* <<*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
Generalisasi/***generalization*** 	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.
Menggunakan/*Include/uses* <<*include*>> 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini

Sumber: (Rosa and Shalahudin, 2018)

2. *Class Diagram*

Menurut (Rosa and Shalahudin, 2018) Unsur-unsur utama dari diagram kelas adalah kotak, yang merupakan ikon yang digunakan untuk mewakili kelas dan *interface*. Setiap kotak dibagi menjadi bagian-bagian horisontal. Bagian atas berisi

nama kelas. Bagian tengah berisi daftar atribut kelas, dan bagian bawah merupakan *operation* dari kelas tersebut. simbol-simbol yang ada pada diagram kelas pada tabel *class diagram* 2.3.

Tabel 2. 2 Simbol *Class Diagram*

Simbol	Deskripsi
Kelas **nama_kelas** +atribut +operasi()	Kelas pada struktur sistem
Antarmuka/*Interface* **nama_interface**	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
Asosiasi/*asociation*	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*
Asosiasi berarah/*directed association*	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya disertai dengan *multiplicity*
Generalisasi	Relasi antar kelas dengan makna generalisasi-spesialisasi(umum khusus)
Kebergantungan/*dependecy*	Relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi/*agregation*	Relasi antar kelas dengan makna semua bagian (*whole-part*)

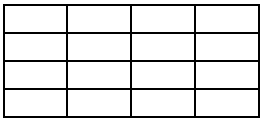
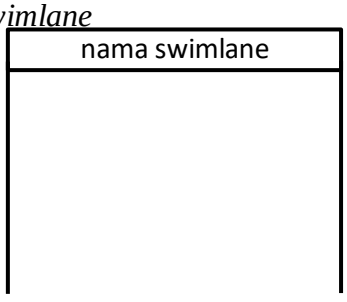
Sumber: (Rosa and Shalahudin, 2018)

3. *Activity Diagram*

Menurut (Rosa and Shalahudin, 2018) Sebuah diagram *activity* menggambarkan perilaku dinamis dari sistem atau bagian dari sistem melalui aliran kontrol antara tindakan yang sistem lakukan. Hal ini mirip dengan sebuah *flowchart*

kecuali bahwa suatu diagram *activity* dapat menunjukkan arus bersamaan. Simbol-simbol yang ada pada *activity diagram* dapat dilihat pada tabel 2.3 di bawah ini :

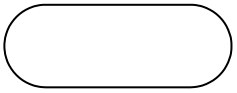
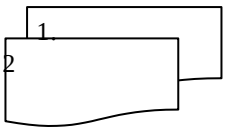
Tabel 2. 3 Simbol *Activity Diagram*

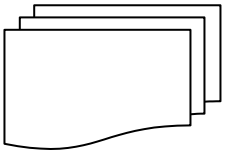
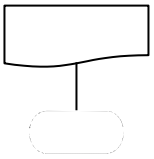
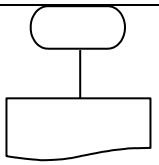
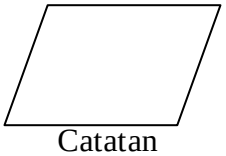
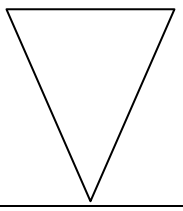
Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem,sebuah diagram aktivitas memiliki sebuah status awal.
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
Percabangan/*decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
Penggabungan/*join*	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
Tabel 	Suatu file komputer dari mana data bisa dibaca atau direkam selama kejadian bisnis
Dokumen 	Menunjukan dokumen sumber atau laporan
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
Swimlane 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

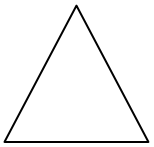
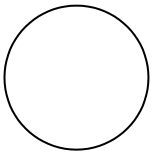
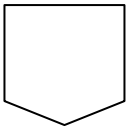
2.8 Bagan Alir Dokumen (*Flowchart*)

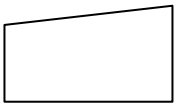
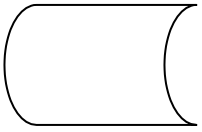
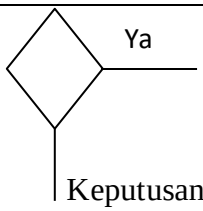
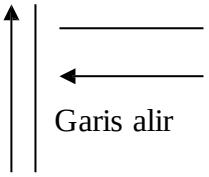
Menurut Jogiyanto (2014) "Bagan alir merupakan bagan yang menunjukkan alir di dalam program atau prosedur sistem secara logika. Bagan alir digunakan terutama untuk alat bantu komunikasi dan untuk dokumentasi". Dapat dilihat pada tabel 2.4 berikut ini :

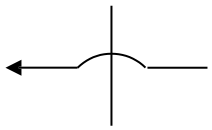
Tabel 2. 4 Simbol Bagan Alir Dokumen

SIMBOL	KETERANGAN
 Mulai/berakhir	Mulai/berakhir (*terminal*), simbol ini untuk menggambarkan awal dan akhir suatu sistem akuntansi.
 Kegiatan manual	Kegiatan manual, simbol ini digunakan untuk menggambrkan kegiatan manual, uraian singkat kegiatan, manual dicantumkan didalam simbol ini.
 Dokumen	Dokumen, digunakan untuk menggambarkan suatu jenis dokumen, yang merupakan formulir yang digunakan untuk merekam data terjadinya suatu transaksi.
	Dokumen dan tembusannya, simbol ini digunakan untuk menggambarkan dokumen asli dan tembusannya. Nomor dokumen dicantumkan disudut kanan atas.

SIMBOL	KETERANGAN
Dokumen dan tembusannya	
 Berbagai dokumen	Berbagai dokumen, simbol ini digunakan untuk menggambarkan berbagai jenis dokumen yang digabungkan bersama didalam satu paket.
 Akhir arus dokumen	Akhir arus dokumen, akhir arus dokumen dan mengarahkan pembaca kesimbol penghubung halaman yang sama yang bernomor seperti yang tercantum didalam simbol tersebut.
 Awal arus dokumen	Awal arus dokumen, awal arus dokumen yang berasal dari simbol penghubung halaman yang sama, bernomor seperti yang tercantum didalam simbol tersebut.
 Catatan	Catatan, simbol ini digunakan untuk menggambarkan catatan akuntansi yang digunakan untuk mencatat data yang direkam sebelumnya didalam dokumen atau formulir.
	Arsip sementara, simbol ini digunakan untuk menunjukkan tempat penyimpanan dokumen, seperti almari arsip dan kotak arsip, terdapat dua tipe arsip yaitu

SIMBOL	KETERANGAN
Arsip sementara	arsip sementara dan arsip permanent. Pengurutan dokumen digunakan simbol sebagai berikut : A = menurut abjad N = menurut nomor urut T = kronologis, menurut tanggal
 *Arsip permanent*	*Arsip permanent*, simbol ini digunakan untuk menggambarkan arsip permanen yang merupakan tempat penyimpanan dokumen yang tidak akan diproses lagi dalam sistem yang bersangkutan.
 Proses	Proses komputer, simbol ini menggambarkan pengolahan data dengan komputer secara *on-line*.
 *on-page connector*	Penghubung pada halaman yang sama (*on-page connector*), karena keterbatasan ruang halaman kertas untuk menggambarkan, maka diperlukan simbol penghubung untuk memungkinkan aliran dokumen berhenti disuatu lokasi lain pada halaman tertentu dan kembali berjalan dilokasi lain pada halaman yang sama.
 *off-page connector*	Penghubung pada halaman yang berbeda (*off-page connector*), jika untuk menggambarkan bagan alir suatu sistem diperlukan lebih dari satu halaman.
	Keterangan, komentar, simbol ini memungkinkan ahli sistem menambahkan keterangan untuk

SIMBOL	KETERANGAN
Keterangan	memperjelaskan pesan yang disampaikan dalam bagan alir.
 Keyboard	Keyboard (*keying*). Simbol ini menggambarkan pemasukan data kedalam komputer melalui *on line* terminal.
 Pita magnetic	Pita magnetic (*magnetic tape*), simbol ini menggambarkan arsip komputer yang berbentuk *pita magnetic.*
 *On-linestorage*	*On-linestorage*, simbol ini menggambarkan arsip komputer yang berbentuk *on-line* (didalam memori komputer).
 Keputusan	Keputusan, simbol ini menggambarkan keputusan yang harus dibuat dalam proses pengolahan data.
 Garis alir	Garis alir (*flowline*), simbol ini menggambarkan keputusan yang harus di buat dalam proses pengolahan data. Anak panah tidak digambarkan jika arus dokumen mengarah kebawah dan kekanan. Jika arus dokumen mengalir keatas atau kekiri, anak panah perlu dicantumkan.

SIMBOL	KETERANGAN
 Persimpangan garis alir	Persimpangan garis alir, jika dua garis alir bersimpangan. Untuk menunjukkan arah masing-masing garis, salah satu garis di buat sedikit melengkung tepat pada persimpangan dua garis tersebut.
 Pertemuan garis alir	Pertemuan garis alir, simbol ini digunakan jika dua garis alir tertentu dan salah satu garis mengikuti arus

2.9 Pengertian SQL

SQL (*Structured Query Language*) adalah bahasa yang digunakan untuk mengelola data pada *Relation* DBMS *(Database Management System)* (Rosa and Shalahudin, 2018). Singkatan dari *Structure Query Language* yang digunakan untuk mendefinisikan struktur data, memodifikasi data pada basis data, menspesifikasi batasan keamanan (*security*), hingga pemeliharaan kinerja basis data (Nugroho, 2015). *SQL* adalah perangkat lunak relation database *management system* (RDBMS) yang didesain untuk melakukan proses manipulasi database berukuran besar dengan berbagai fasilitas (Kristanto, 2018). Jadi *Structure Query Language*adalah perangkat lunak relation database*management system* (RDBMS) mendefinisikan struktur data, memodifikasi data pada basis data, menspesifikasi batasan keamanan (*security*), hingga pemeliharaan kinerja basis data.

2.10 Javascript

JavaScript dimulai sekitar tahun 1994, pada tahun 1995, Brendan Eich mulai mengembangkan sebuah bahasa pemrograman script dinamakan Mocha. Bahasa Mocha ditujukan untuk client side dan juga server side. JavaScript merupakan bahasascript berbasis objek yang mengijinkanpenguna untuk mengendalikan banyak aspek interaksi penguna pada dokumen HTML. Semua objet tersebut memiliki properti yang saling berhubungan dengannya (Saifudin & Setiaji, 2019).

2.11 Android

Android adalah sistem operasi untuk telpon seluler berbasis Linux sebagai karnelnya. *Android* menyediakan *platform* terbuka (*open source*) bagi *developer* untuk menciptakan aplikasinya sendiri. *Android* Inc. adalah pendatang baru yang bergerak dalam pembuatan piranti lunak untuk ponsel. (Sasmito and Hadiansah, 2015).

Android adalah *mobile Operating System* (OS) yang dikembangkan oleh Google. OS *Android* berbasis pada OS Linux Kernel. *Android* bersifat open source, artinya pengembang bisa memodifikasi dan menyesuaikan OS untuk setiap ponsel. *Android* menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk di gunakan oleh bermacam-macam kegunaan. *Android* beberapa kali melakukan pembaruan versinya, kebanyakan nama di setiap versinya adalah nama makanan.

Gambar 2. 2 Logo *Android*

Android memiliki empat karakteristik sebagai berikut :

1. Terbuka

   Android dibangun untuk benar-benar terbuka sehingga sebuah aplikasi dapat memanggil salah satu fungsi inti ponsel seperti membuat panggilan, mengirim pesan teks, menggunakan kamera dan lain-lain. *Android* merupakan sebuah mesin virtual yang dirancang khusus untuk mengoptimalkan sumber daya memori dan perangkat keras yang terdapat di dalam perangkat. *Android* merupakan *open source*, dapat secara bebas diperluas untuk memasukkan teknologi baru yang lebih maju pada saat teknologi tersebut muncul. *Platform* ini akan terus berkembang untuk membangun aplikasi *mobile* yang inovatif.

2. Semua aplikasi dibuat sama

   Android tidak memberikan perbedaan terhadap aplikasi utama dari telepon dan aplikasi pihak ketiga (*third-party application*). Semua apliksi dapat dibangun untuk memiliki akses yang sama terhadap kemampuan sebuah telepon dalam menyediakan layanan dan aplikasi yang luas terhadap para pengguna.

3. Memecahkan hambatan pada aplikasi

Android memecah hambatan untuk membangun aplikasi yang baru dan inovatif. Misalnya, pengembang dpat menggabungkan informasi yang diperoleh dari *web* dengan data pada ponsel seseorang seperti kontak pengguna, kalender atau lokasi geografis.

4. Pengembangan aplikasi yang cepat dan mudah

   Android menyediakan akses yang sangat luas kepada pengguna untuk menggunakan aplikasi yang semakin baik. *Android* memiliki sekumpulan tools yang dapat digunakan sehingga membantu para pengembang dalam meningkatkan produktivitas pada saat membangun aplikasi yang dibuat.

Versi *Android* diawali dengan dirilisnya *Android* beta pada bulan November 2007. Komersial pertama, *Android* 1.0, dirilis pada September 2008. Sejak April 2009, versi *Android* dikembangkan dengan nama kode yang dinamai berdasarkan makanan pencuci mulut dan makanan manis. Masing-masing versi dirilis sesuai urutan alfabet, yaitu :

1. *Cupcake (1.5)*
2. *Donut (1.6)*
3. *Eclair (2.0–2.1)*
4. *Froyo (2.2–2.2.3)*
5. *Gingerbread (2.3–2.3.7)*
6. *Honeycomb (3.0–3.2.6)*
7. *Ice Cream Sandwich (4.0–4.0.4)*
8. *Jelly Bean (4.1–4.3)*
9. *KitKat (4.4+)*
10. *Lollipop (5.0-5.1)*

11. *Marshmallow (6.0)*
12. *Nougat (7.0)*
13. *Oreo (8.0)*
14. *Pie (9.0)*
15. *Android 10.*

2.12 ISO 9126

Salah satu tolak ukur kualitas perangkat lunak adalah ISO 9126, yang dibuat oleh *International Organization for Standardization (ISO)* dan *International Electrotechnical Commission (IEC)*. ISO 9126 mendefinisikan kualitas produk perangkat lunak, model, karakteristik mutu, dan metrik terkait yang digunakan untuk mengevaluasi dan menetapkan kualitas sebuah produk software. Standar ISO 9126 telah dikembangkan dalam usaha untuk mengidentifikasi atribut-atribut kunci kualitas untuk perangkat lunak komputer. Menurut (Abran *et al.*, 2018) ISO 9126 adalah standar internasional yang diterbitkan oleh ISO untuk evaluasi kualitas perangkat lunak dan merupakan pengembangan dari ISO 9001. Faktor kualitas menurut ISO 9126 meliputi enam karakteristik kualitas sebagai berikut.

1) *Functionality* (Fungsionalitas). Kemampuan perangkat lunak untukmenyediakan fungsi sesuai kebutuhan pengguna, ketika digunakan dalam kondisi tertentu.
2) *Reliability* (Kehandalan). Kemampuan perangkat lunak untuk mempertahankan tingkat kinerja tertentu, ketika digunakan dalam kondisi tertentu.

3) *Usability* (Kebergunaan). Kemampuan perangkat lunak untuk dipahami, dipelajari, digunakan, dan menarik bagi pengguna, ketika digunakan dalam kondisi tertentu.

4) *Efficiency* (Efisiensi). Kemampuan perangkat lunak untuk memberikan kinerja yang sesuai dan relatif terhadap jumlah sumber daya yang digunakan pada saat keadaan tersebut.

5) *Maintainability* (Pemeliharaan). Kemampuan perangkat lunak untuk dimodifikasi.Modifikasi meliputi koreksi, perbaikan atau adaptasi terhadap perubahan lingkungan, persyaratan, dan spesifikasi fungsional.

6) *Portability* (Portabilitas). Kemampuan perangkat lunak untuk di *transfer* dari satu lingkungan ke lingkungan lain.

Masing-masing karakteristik kualitas perangkat lunak model ISO 9126 dibagi menjadi beberapa sub-karakteristik kualitas. Berikut adalah tabel karakteristik Kualitas Perangkat Lunak Model ISO 9126 :

Tabel 2. 5 Karakteristik ISO 9126

Karakteristik	**Sub Karakteristik**	**Deskripsi**
Functionality	*Suitability*	Kemampuan perangkat lunak untuk menyediakan serangkaian fungsi yang sesuai untuk tugas-tugas tertentu dan tujuan pengguna.
	Accuracy	Kemampuan perangkat lunak dalam memberikan hasil yang presisi dan benar sesuai dengan kebutuhan.

Karakteristik	Sub Karakteristik	Deskripsi
	Security	Kemampuan perangkat lunak untuk mencegah akses yang tidak diinginkan, menghadapi penyusup (*hacker*) maupun otorisasi dalam modifikasi data.
	Interoperabiy	Kemampuan perangkat lunak untuk berinteraksi dengan satu atau lebih sistem tertentu.
	Compliance	Kemampuan perangkat lunak dalam memenuhi standar dan kebutuhan sesuai peraturan yang berlaku.
Reliability	*Maturity*	Kemampuan perangkat lunak untuk menghindari kegagalan sebagai akibat dari kesalahan dalam perangkat lunak.
	Fault tolerance	Kemampuan perangkat lunak untuk mempertahankan kinerjanya jika terjadi kesalahan perangkat lunak
	Recoverabiliy	Kemampuan perangkat lunak untuk membangun kembali tingkat kinerja ketika terjadi kegagalan sistem, termasuk data dan koneksi jaringan.
Usability	*Understandibity*	Kemampuan perangkat lunak dalam kemudahan untuk dipahami.
	Learnability	Kemampuan perangkat lunak dalam kemudahan untuk dipelajari.

Karakteristik	Sub Karakteristik	Deskripsi
	Operability	Kemampuan perangkat lunak dalam kemudahan untuk dioperasikan.
	Attractiveness	Kemampuan perangkat lunak dalam menarik pengguna.
Efficiency	*Time behavior*	Kemampuan perangkat lunak dalam memberikan respon dan waktu pengolahan yang sesuai saat melakukan fungsinya.
	Resource behavior	Kemampuan perangkat lunak dalam menggunakan sumber daya yang dimilikinya ketika melakukan fungsi yang ditentukan.
Maintainabiliy	*Analyzability*	Kemampuan perangkat lunak dalam mendiagnosis kekurangan atau penyebab kegagalan.
	Changeabiliy	Kemampuan perangkat lunak untuk dimodifikasi tertentu.
	Stability	Kemampuan perangkat lunak untuk meminimalkan efek tak terduga dari modifikasi perangkat lunak.
	Testability	Kemampuan perangkat lunak untuk dimodifikasi dan divalidasi perangkat lunak lain.

Karakteristik	Sub Karakteristik	Deskripsi
Portability	*Adaptability*	Kemampuan perangkat lunak untuk diadaptasikan pada lingkungan yang berbeda-beda.
	Instalability	Kemampuan perangkat lunak untuk diinstal dalam lingkungan yang berbeda-beda.
	Coexistence	Kemampuan perangkat lunak untuk berdampingan dengan perangkat lunak lainnya dalam satu lingkungan dengan berbagi sumber daya.
	Replaceabiliy	Kemampuan perangkat lunak untuk digunakan sebagai sebagai pengganti perangkat lunak lainnya.

Sumber: (Al-Qutaish 2010, 172-173)

Adapun alasan penggunaan ISO 9126 karena ISO sudah berstandar *International Organization for Standardization (ISO)* dan *International Electrotechnical Commission (IEC).* Kualitas produk perangkat lunak ISO 9126 memiliki enam karakteristik pendukung yang dapat digunakan sebagai acuan dalam menilai maupun memberikan masukan terhadap kualitas perangkat lunak yang akan dibangun yang akan menghasilkan nilai uji yang terukur. Indikator yang digunakan dalam pengujian ISO 9126 dilihat dari sisi *Functionality, Usability*, dan *Reability.*