

BAB II

TINJAUAN PUSTAKA

1.1 E-Commerce

Definisi *E-Commerce* Laudon and Laudon (2013), adalah media perdagangan elektronik yang memiliki karakteristik atau sifat-sifat tertentu. Berdasarkan sifat penggunaannya menurut para ahli ini, *E-Commerce* dapat dibagi menjadi beberapa tipe atau jenis, antara lain:

1. *Business-to-consumer* (B2C)

B2C adalah kegiatan *e-business* dalam pelayanan secara langsung kepada konsumen melalui barang atau jasa. Dengan melakukan transaksi penjualan secara langsung dan pemesanan dapat langsung dilakukan oleh konsumen karena biaya sudah tercantum, berikut kelebihan B2C yaitu disebut dengan transaksi pasar, konsumen mempelajari produk yang ditawarkan melalui publikasi, membeli dengan elektronik cash dan sistem pembayaran transfer dan adanya permintaan pengiriman barang.

2. *Business-to-business* (B2B)

B2B adalah transaksi secara elektronik antara entitas atau objek yang satu ke objek bisnis lainnya, dapat disimpulkan B2B adalah transaksi antar perusahaan, transaksi pembelian jasa maupun barang dan pertukaran dokumen bisnis antar aplikasi komputer, antara instansi secara elektronik menggunakan format standar yang telah disepakati.

3. *Consumer-to-consumer* (C2C)

Consumer-to-consumer (C2C) *E-Commerce* merupakan tipe yang paling relevan dengan pembahasan dalam paper ini. *E-Commerce* atau perdagangan elektronik C2C merujuk pada transaksi finansial maupun informational yang dilakukan langsung antar konsumen. Sedangkan *E-Commerce* C2C memungkinkan konsumen untuk menjual produknya (barang atau jasa) langsung kepada konsumen lain yang pada umumnya dipertemukan melalui situs bisnis tertentu.

Pada penelitian ini akan berfokus pada B2C pada aplikasi e-commerce yaitu transaksi yang dilakukan oleh perusahaan dan ditujukan kepada konsumen, sehingga perusahaan selaku penyedia barang melakukan penjualan untuk konsumen.

1.2 Rumah Jamur

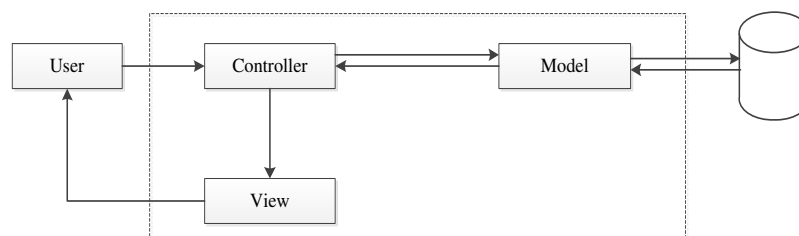
Menurut Susilawati and Budi (2018), Rumah jamur atau dengan istilah lain adalah kumbung merupakan tempat menyimpan media tanam agar pertumbuhan jamur dapat tumbuh dengan baik dan menghasilkan jamur yang berkualitas (baik dari segi berat dan bentuk). Bahan untuk membuat rumah jamur dari bahan yang mudah didapat disekitar lokasi, murah dan kuat.

Menurut Hendro (2015), Kumbung adalah bangunan yang dibuat untuk keperluan budidaya jamur. Tujuan dibangun kumbung adalah untuk melindungi baglog jamur dari hujan, sinar matahari langsung, dan kemungkinan kontaminasi spora jamur. Disamping itu, kumbung juga berguna untuk merekayasa kondisi iklim secara mikro di dalam ruangan kumbung, sehingga budidaya jamur yang dilakukan tidak tergantung kondisi musim dan cuaca di daerah sekitar. Dengan adanya bangunan kumbung kita bisa merekayasa kondisi suhu dan kelembaban yang kita inginkan.

1.3 CodeIgniter

Menurut Raharjo (2018), *CodeIgniter* adalah *Framework* untuk bahasa pemrograman PHP, yang dibuat Rick Ellis pada tahun 2006. *CodeIgniter* memiliki banyak fitur yang membantu para pengembang PHP untuk dapat membuat aplikasi secara mudah dan cepat serta memiliki sifat yang fleksibel dapat mengembangkan dalam perangkat *web*, dekstop maupun *mobile*”.

CodeIgniter memiliki konsep atau pola *Model-View-Controller* (MVC) sehingga kode-kode dapat di sederhanakan.



Gambar 1.1 Arsitektur MVC

1.3.1 Web Based

Menurut Urbieta *et al* (2019), *Web Based* adalah aplikasi yang dibuat berbasis *web* yang membutuhkan *web server* dan *browser* untuk menjalankannya. Dengan membuat sistem berbasis

web based ada beberapa hal yang penting dan harus kita pikirkan sebelum membangun sistem tersebut, diantaranya:

1. Tidak membutuhkan *hardware* dengan spesifikasi yang tangguh untuk menjalankan aplikasinya.
2. Server yang dibutuhkan cukup diinstallkan *tools* pendukung saja agar klien mudah menjalankan aplikasi
3. Infrastruktur jaringan yang dibutuhkan juga cukup besar karena aplikasi yang dibuat dapat diakses dari jaringan luar (internet).
4. Aplikasi berbasis *web based* dapat diakses dari berbagai perangkat dengan syarat menggunakan *web browser* saja sudah dapat mengaksesnya.
5. Jika aplikasi yang sudah jadi ingin di *update*, sangat mudah untuk melakukannya karena tidak membutuhkan membuka keseluruhan aplikasi.

1.3.2 PHP

Menurut Subagja (2018), PHP adalah bahasa *server-side-scripting* yang menyatudengan HTML untuk membuat halaman *web* yang dinamis. Menurut (Aryani, Setiadi and Alfiah, 2015), berpendapat bahwa *PHP Hypertext Preprocessor* adalah bahasa pemrograman *web server-side* yang bersifat *open source*. PHP merupakan *script* yang terintegrasi dengan HTML dan berada pada *server* (*server side HTML embedded scripting*). PHP adalah *script* yang digunakan untuk membuat halaman *website* yang dinamis.

Dinamis berarti halaman yang akan ditampilkan dibuat saat halaman itu diminta oleh *client*. Mekanisme ini menyebabkan informasi yang diterima *client* selalu yang terbaru/*up to date*. Semua *script* PHP dieksekusi pada *server* dimana *script* tersebut dijalankan. Dengan menggunakan program PHP, sebuah *website* akan lebih interaktif dan dinamis.

Sehingga PHP merupakan bahasa pemrograman yang digunakan oleh pengembang untuk membuat sistem *website* dengan kumpulan bahasa HTML dan *script* lainnya.

1.3.3 MySql

Menurut MySQL (2018), *MySQL* adalah singkatan dari *Structue Query Language* yang digunakan untuk mendefinisikan structure data, memodifikasi data pada basis data, menspesifikasi batasan keamanan (*security*), hingga pemeliharaan data.

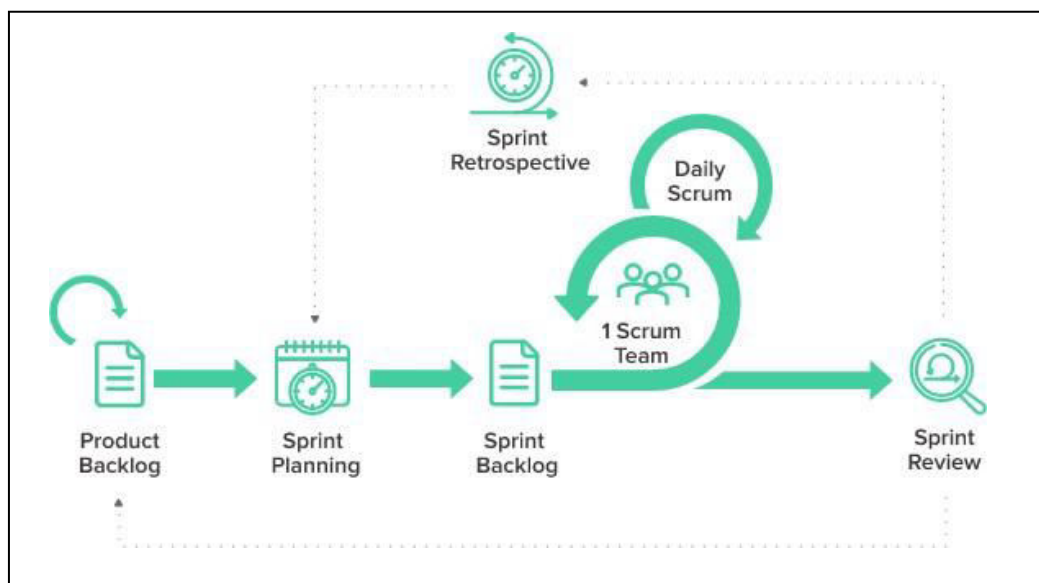
Menurut Amin (2018), mendefinisikan *mysql* adalah RDBMS yang cepat dan mudah digunakan, serta sudah banyak digunakan untuk berbagai kebutuhan.

MySQL merupakan bahasa standar yang paling banyak digunakan untuk mengakses *database* relasional dan merupakan aplikasi yang dapat dipergunakan secara bebas.

1.4 Metode Scrum

Menurut Adi & Permana (2015) dalam penelitiannya yang berjudul “*Scrum Method Implementation in a Software Development Project Management* ” dalam (Julianto, 2019) menyatakan bahwa *Scrum* pertama kali dikembangkan oleh Schwaber dan Sutherland pada tahun 1993 dan tujuannya adalah menjadi metodologi pengembangan yang mengikuti prinsip-prinsip metodologi *Agile*.

Menurut (Schwaber & Sutherland, 2017) *Scrum* adalah suatu metodologi atau kerangka kerja yang terstruktur untuk mendukung pengembangan produk yang kompleks. *Scrum* terdiri dari sebuah tim yang memiliki peran dan tugas masing- masing. Setiap komponen dalam kerangka melayani tujuan tertentu dan sangat penting untuk kesuksesan penggunaan *Scrum*.



Gambar 1.2 Tahapan-Tahapan Metode *Scrum*

1.4.1 Tahapan-Tahapan Metode *Scrum*

Adapun tahapan-tahapan dalam *Scrum* dalam (Schwaber & Sutherland, 2017) adalah sebagai berikut ini:

1. *Product Backlog*

Peneliti sistem akan mengumpulkan dan menyusun semua kebutuhan sistem dan permintaan pengguna terhadap sistem, misalnya fitur-fitur yang dibutuhkan oleh pengguna terhadap sistem. *Product backlog* berada dalam tanggung jawab *product owner*. Setelah targetnya ditetapkan, semua kebutuhan dan permintaan akan dibagikan menjadi poin-poin kecil yang mana setiap poin tersebut mempunyai tingkat layak untuk dikembangkan.

2. *Sprint Planning*

Sprint Planning merupakan sebuah langkah yang wajib dilaksanakan setiap saat akan memulainya sprint baru. Pada langkah tersebut peneliti akan menyusun pekerjaan-pekerjaan apa saja yang harus diselesaikan dalam 1 *sprint*.

3. *Sprint Backlog*

Perencanaan *sprint* dilakukan dalam pertemuan/*meeting* antara pemilik produk dan tim developer, yang akan berkolaborasi untuk memilih produk yang akan dikembangkan *backlog* untuk dimasukkan kedalam proses *sprint*. Hasil dari pertemuan tersebut adalah *sprint backlog*.

4. *Sprint*

Dalam *Scrum*, *Sprint* adalah sebuah kerangka waktu yang berdurasi maksimal 1 bulan untuk mengembangkan produk yang berpotensi untuk dirilis. Dalam *Sprint* terdapat 2 bagian pekerjaan, yaitu:

a. *Pertemuan Harian (Daily Standup Meeting)*

Merupakan pertemuan dimana setiap 24 jam (1 hari), tim pengembang bertemu untuk membahas proses pengembangan produk.

b. *Refleksi Sprint*

Merupakan pertemuan yang dilakukan setiap bulannya, yang bertujuan untuk membahas hal dari *Sprint Backlog* yang telah berjalan dan telah berhasil dikerjakan, serta dapat

memperbaiki dan meningkatkan kualitas produk pada *Sprint* yang berikutnya.

5. Working Increment (Sprint Rivew)

Increment merupakan hasil dari seluruh hal dalam *product backlog* yang telah selesai dikerjakan pada seluruh *sprint*.

1.4.2 Kelebihan Metode *Scrum*

Kelebihan metode *scrum* diantaranya adalah;

1. *Scrum* memberikan kepuasan pelanggan dengan
2. Mengoptimalkan waktupenyelesaian dan responsif terhadap permintaan.
3. Meningkatkan kualitas.
4. Terima dan harapkan perubahan.
5. Memberikan perkiraan yang lebih baik sambil menghabiskan lebih sedikitwaktu untuk tahap pengembangan.

1.5 Alat Pengembang Sistem


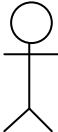

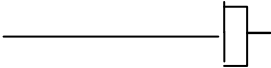
Alat pengembang sistem merupakan konsep desain yang digunakan untuk menggambarkan sistem dengan menggunakan diagram. Penyesuaian alat yang digunakan harus sesuai dengan metode pengembangan yang dilakukan salah satunya adalah penerapan *Unified Modelling Language*. Menurut Rosa dan Salahuddin (2019), *Unified Modelling Language* adalah bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung. Berikut ini merupakan penjelasan tentang masing-masing diagram yang ada pada *Unified Modelling Language*.

1.5.1 Use Case Diagram



Menurut Rosa dan Salahuddin (2019), *use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Berikut simbol-simbol yang akan digunakan dalam menggambarkan *Use Case Diagram* dapat dilihat pada Tabel 2.1 :

Tabel 1.1 Simbol *Use Case Diagram*

No	Simbol	Deskripsi
----	--------	-----------

1.		<i>Use case</i> : Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal <i>frase</i> nama <i>use case</i> .
2.		Aktor: seseorang/sesuatu yang berinteraksi dengan yang akan dibuat. diluar sistem informasi. Biasanya dinyatakan menggunakan kata benda
3.		Asosiasi (<i>association</i>): merupakan komunikasi antara aktor dan <i>use case</i> yang berpartisipasi pada <i>use case</i> atau <i>use case</i> memiliki interaksi dengan aktor.
4.		Generalisasi (<i>generalization</i>): merupakan hubungan (umum – khusus) antara dua buah <i>use case</i> dimana fungsi yang satu adalah fungsi yang lebih umum

Tabel 2.1 Simbol *Use Case Diagram* (Lanjutan)

No	Simbol	Deskripsi
5.	<< Include >> 	Include berarti <i>use case</i> yang ditambahkan akan dipanggil saat <i>use case</i> tambahan dijalankan.
6.	<<Extend>> 	Ekstensi (<i>extend</i>) merupakan <i>use case</i> tambahan ke sebuah <i>use case</i> yang ditambahkan dapat berdiri sendiri walau tanpa <i>use case</i> tambahan itu.

Sumber: (Rosa dan Salahuddin, 2019)

1.5.2 *Class Diagram*

Menurut Rosa dan Salahuddin (2019), *Class diagram* mengembangkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Berikut simbol-simbol yang akan digunakan dalam menggambarkan *Class Diagram* dapat dilihat pada tabel 2.3:

Tabel 1.2 Simbol *Class Diagram*

No.	Simbol	Deskripsi
1.		Kelas pada struktur sistem.
2.	<p>Antar Muka/Interface</p>  <p>Nama_Interface</p>	Sama dengan konsep interface dalam pemrograman berorientasi objek.
3.	<p>Asosiasi / Asociation</p> 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan symbol
4.	<p>Asosiasi Berarah / Directed Association</p> 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan symbol.
5.	<p>Generalisasi</p> 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum khusus)
6.	<p>Ketergantungan / dependency</p> 	Relasi antar kelas dengan makna ketergantungan antar kelas.
7.	<p>Agregasi / aggregation</p> 	Relasi antar kelas dengan maksna semua bagian (<i>whole-part</i>)

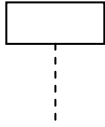

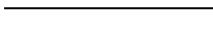
Sumber: (Rosa dan Salahuddin, 2019)

1.5.3 *Sequence Diagram*

Diagram rangkaian menggambarkan bagaimana objek berinteraksi dengan satu sama lain melalui pesan pada eksekusi sebuah use-case atau operasi. Diagram ini mengilustrasikan bagaimana pesan terkirim dan diterima di antara objek dan dalam sekuensi.

Tabel 1.3 Simbol *Sequence Diagram*

No.	Simbol	Deskripsi
-----	--------	-----------

1.	<p style="text-align: center;"><i>Object lifeline</i></p> 	Menggambarkan panjang kehidupan suatu objek selama scenario sedang di buat contohnya
2.	<p style="text-align: center;"><i>Activation</i></p> 	Dimana proses sedang dilakukan oleh <i>object</i> atau <i>class</i> untuk memenuhi pesan atau perintah
3.	<p style="text-align: center;"><i>Message</i></p> 	Sebuah anak panah yang mengindikasikan pesan diantara objek. Dan objek dapat mengirimkan pesan ke dirinya sendiri


Sumber: (Rosa dan Salahuddin, 2019)


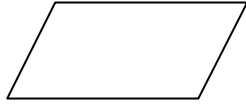
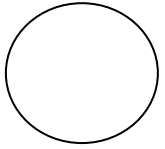
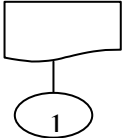
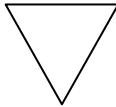
1.6 BAD (*Document Flowchart*)

Dengan menggunakan *flowchart*, langkah prosedur penyelesaian permasalahan dapat diekspresikan dengan serangkaian simbol grafis yang baku dan lebih mudah digunakan. Manfaat yang diperoleh menggunakan *flowchart* yaitu membiasakan berfikir secara sistematis dan terstruktur dalam setiap kesempatan serta lebih mudah mengecek dan menemukan bagian-bagian prosedur yang tidak valid.

Menurut (Nimas, 2016) *flowchart* adalah bagan (*chart*) yang menunjukkan alir (*flow*) di dalam program atau prosedur sistem secara logika. Simbol – simbol Bagan Alir Dokumen seperti berikut :

Tabel 1.4 Simbol – simbol Bagan Alir Dokumen

Simbol	Keterangan
 <p style="text-align: center;">Dokumen</p>	Simbol ini menggambarkan semua jenis dokumen yang merupakan formulir yang digunakan untuk merekam data terjadinya suatu transaksi.

 <p>Dokumen dan Tembusannya</p>	<p>Simbol ini digunakan untuk menggambarkan dokumen asli dan tembusannya. Nomor lembar dokumen dicantumkan di sudut kanan atas.</p>
 <p>Catatan</p>	<p>Simbol ini digunakan untuk menggambarkan catatan akuntansi yang digunakan untuk mencatat data yang direkam sebelumnya dalam dokumen atau formulir.</p>
 <p>Connector</p>	<p>Simbol ini sebagai tanda penghubung bagan alir dokumen pada halaman yang sama, dengan memperhatikan nomor yang tercantum dalam simbol penghubung pada halaman yang sama.</p>
 <p>Awal arus dokumen</p>	<p>Berasal dari simbol penghubung halaman yang sama, yang bernomor seperti yang tercantum di dalam simbol tersebut.</p>
 <p>Arsip Sementara</p>	<p>Simbol ini digunakan untuk menunjukkan tempat penyimpanan dokumen seperti lemari arsip dan kotak arsip. Keterangan: A = Menurut abjad N = Menurut nomor urut T = Menurut tanggal / kronologis</p>

1.7 Metode Pengujian Sistem

Metode pengujian sistem merupakan metode yang digunakan untuk melakukan testing pada sistem yang dibangun sehingga di peroleh hasil berupa sistem yang sesuai fungsinya.

1.7.1 ISO 25010

ISO/IEC 25010 merupakan model kualitas sistem dan perangkat lunak yang menggantikan ISO/IEC 9126 tentang software engineering (International Organisation for Standardisation, 2011). Product quality ini juga digunakan untuk tiga model kualitas yang berbeda untuk produk perangkat lunak antara lain:

1. Kualitas dalam model penggunaan,
2. Model kualitas produk, dan
3. Data model kualitas

Model kualitas produk terdiri dari delapan karakteristik yang berhubungan dengan sifat statis perangkat lunak dan sifat dinamis dari sistem komputer. Model ini berlaku untuk sistem komputer dan produk perangkat lunak. Karakteristik yang didefinisikan oleh kedua model tersebut relevan untuk semua produk perangkat lunak dan sistem komputer. Karakteristik dan subkarakteristik memberikan terminologi yang konsisten untuk menentukan, mengukur dan mengevaluasi kualitas sistem dan perangkat lunak. Mereka juga menyediakan seperangkat karakteristik kualitas yang sesuai dengan persyaratan kualitas yang dapat dibandingkan untuk kelengkapan.

1.7.2 Tahapan Pengujian Sistem

Tahapan pengujian sistem digunakan untuk mengetahui proses pengujian yang akan dilakukan, berikut adalah tahapan pengujian ISO:

1. *Functional Suitability*

Sejauh mana perangkat lunak mampu menyediakan fungsi yang memenuhi kebutuhan yang dapat digunakan dalam kondisi tertentu. Karakteristik ini dibagi menjadi beberapa karakteristik yaitu.

- a. *Functional completeness*, sejauh mana fungsi yang disediakan mencakup semua tugas dan tujuan pengguna secara spesifik.
- b. *Functional correctness*, sejauh mana produk atau sistem menyediakan hasil yang benar sesuai kebutuhan.
- c. *Functional appropriateness*, sejauh mana fungsi yang disediakan mampu memfasilitasi penyelesaian tugas dan tujuan tertentu.

2. *Compatibility*

Sejauh mana sebuah produk, sistem atau komponen dapat bertukar informasi dengan produk, sistem atau komponen dan/atau menjalankan fungsi lain yang diperlukan secara bersamaan ketika berbagi perangkat keras dan environment perangkat lunak yang sama. Karakteristik ini dibagi menjadi 2 karakteristik yaitu.

- a. *Co-existence*, sejauh mana produk atau sistem dapat menjalankan fungsi yang dibutuhkan secara efisien sementara berbagi sumber daya dengan produk atau sistem yang lain tanpa merugikan produk atau sistem tersebut.

b. *Interoperability*, sejauh mana dua atau lebih produk, sistem atau komponendapat bertukar informasi dan menggunakan informasi tersebut.

3. *Usability*

Sejauh mana sebuah produk atau sistem dapat digunakan oleh user tertentu untuk mencapai tujuan dengan efektif, efisiensi, dan kepuasan tertentu dalam konteks penggunaan. Karakteristik ini terbagi menjadi beberapa karakteristik yaitu.

a. *Appropriateness recognizability*, sejauh mana pengguna dapat mengetahui apakah sistem atau produk sesuai kebutuhan mereka.

b. *Learnability*, sejauh mana produk atau sistem dapat digunakan oleh pengguna untuk mencapai tujuan tertentu yang belajar menggunakan sistem atau produk dengan efisien, efektif, kebebasan dari resiko dan kepuasan dalam konteks tertentu.

c. *Operability*, sejauh mana produk atau sistem mudah dioperasikan dan dikontrol.

d. *User error protection*, sejauh mana produk atau sistem melindungi pengguna terhadap membuat kesalahan.

e. *User interface aesthetics*, sejauh mana antarmuka pengguna dari produk atau sistem memungkinkan interaksi yang menyenangkan dan memuaskan pengguna.

f. *Accessibility*, sejauh mana produk atau sistem dapat digunakan oleh semua kalangan untuk mencapai tujuan tertentu sesuai konteks penggunaan.

4. *Reliability*

Sejauh mana sebuah sistem, produk atau komponen dapat menjalankan fungsi tertentu dalam kondisi tertentu selama jangka waktu yang ditentukan.

Karakteristik ini terbagi menjadi beberapa subkarakteristik yaitu.

a. *Maturity*, sejauh mana produk atau sistem mampu memenuhi kebutuhan secara handal di bawah keadaan normal.

b. *Availability*, sejauh mana produk atau sistem siap beroperasi dan dapat diakses saat perlu digunakan.

- c. *Fault tolerance*, sejauh mana produk atau sistem tetap berjalan sebagaimana yang dimaksud meskipun terjadi kesalahan pada perangkat keras atau perangkat lunak.
- d. *Recoverability*, sejauh mana produk atau sistem mampu dapat memulihkan data yang terkena dampak secara langsung dan menata ulang kondisi system seperti yang diinginkan ketika terjadi gangguan.

5. *Security*

Sejauh mana sebuah produk atau sistem melindungi informasi dan data sehingga seseorang atau sistem lain dapat mengakses data sesuai dengan jenis dan level otorisasi yang dimiliki. Karakteristik ini terbagi menjadi beberapa karakteristik yaitu.

- a. *Confidentiality*, sejauh mana produk atau perangkat lunak memastikan data hanya bisa diakses oleh mereka yang berwenang untuk memiliki akses.
- b. *Integrity*, sejauh mana produk atau perangkat lunak mampu mencegah akses yang tidak sah untuk memodifikasi data.
- c. *Non-repudiation*, sejauh mana peristiwa atau tindakan dapat dibuktikan telah terjadi, sehingga tidak ada penolakan terhadap peristiwa atau tindakan tersebut.
- d. *Accountability*, sejauh mana tindakan dari suatu entitas dapat ditelusuri secara unik untuk entitas.
- e. *Authenticity*, sejauh mana identitas subjek atau sumber daya dapat terbukti menjadi salah satu yang diklaim.

6. *Portability*

Sejauh mana keefektifan dan efisiensi sebuah sistem, produk atau komponen dapat dipindahkan dari satu perangkat keras, perangkat lunak atau digunakan pada lingkungan yang berbeda. Karakteristik ini dibagi menjadi beberapa karakteristik yaitu.

- a. *Adaptability*, sejauh mana produk atau sistem dapat secara efektif dan efisien disesuaikan pada perangkat lunak, perangkat keras dan lingkungan yang berbeda.
- b. *Installability*, sejauh mana produk atau sistem dapat berhasil dipasang atau dihapus dalam lingkungan tertentu.
- c. *Replaceability*, sejauh mana produk atau sistem dapat menggantikan produk atau sistem lain yang ditentukan untuk tujuan yang sama pada lingkungan yang sama.

7. *Performance Efficiency*

Kinerja relatif terhadap sumber daya yang digunakan dalam kondisi tertentu. Karakteristik ini terbagi menjadi beberapa subkarakteristik yaitu.

- a. *Time behaviour*, sejauh mana respon dan pengolahan waktu produk atau sistem dapat memenuhi persyaratan ketika menjalankan fungsi.
- b. *Resource utilization*, sejauh mana jumlah dan jenis sumber daya yang digunakan oleh produk atau sistem dapat memenuhi persyaratan ketika menjalankan fungsi.
- c. *Capacity*, sejauh mana batas maksimum parameter produk atau sistem dapat memenuhi persyaratan.

8. *Maintainability*

Sejauh mana keefektifan dan efisiensi dari sebuah produk atau sistem dapat dirawat. Karakteristik ini terbagi menjadi beberapa subkarakteristik yaitu.

- a. *Modularity*, sejauh mana sistem terdiri dari komponen terpisah sehingga perubahan atau modifikasi pada salah satu komponen tersebut memiliki dampak yang kecil terhadap komponen yang lain.
- b. *Reusability*, sejauh mana aset dapat digunakan lebih oleh satu sistem atau digunakan untuk membangun aset lain.
- c. *Analyzability*, tingkat efektivitas dan efisiensi untuk mengkaji dampak perubahan pada satu atau lebih bagian-bagian produk atau sistem, untuk mendiagnosis kekurangan atau penyebab kegagalan produk, untuk mengidentifikasi bagian yang akan diubah.
- d. *Modifiability*, sejauh mana produk atau sistem dapat dimodifikasi secara efektif dan efisien tanpa menurunkan kualitas produk yang ada.
- e. *Testability*, tingkat efektivitas dan efisiensi untuk membentuk kriteria uji dari produk, sistem atau komponen dan uji dapat dilakukan untuk menentukan apakah kriteria tersebut telah terpenuhi