BAB 2

TINJAUAN REFERENSI

2.1 Internet

Kata Internet berasal dari "*International Network*" (William dan Sawyer, 2010, p. 60). Internet merupakan jantung dari Era Informasi. Internet disebut juga dengan sebutan "ibu dari semua jaringan". Internet sendiri memiliki definisi yaitu jaringan komputer di seluruh dunia yang menghubungkan ratusan ribu jaringan kecil lainnya. Jaringan-jaringan ini menghubungkan entitas pendidikan, komersial, nirlaba, militer, dan juga individu. Internet sudah ada sejak lebih dari 40 tahun yang lalu. Namun, yang membuat Internet menjadi populer, selain email, adalah pengembangan *World Wide Web* di awal tahun 1990. (William dan Sawyer, 2010, p. 18).

2.1.1 *World Wide Web* (WWW)

World Wide Web, atau sering disebut hanya dengan "*Web*", adalah sebuah sistem yang saling berhubungan dari komputer internet (*server*) yang mendukung dokumen-dokumen yang diformat dengan khusus dalam wujud multimedia. Kata "multimedia" sendiri berasal dari "*multiple media*", mengacu kepada teknologi yang menyampaikan informasi dalam lebih dari satu media, seperti teks, gambar diam, gambar bergerak, dan suara. Dapat disimpulkan juga, bahwa *web* menyediakan informasi dalam lebih dari satu cara. (William dan Sawyer, 2010, p. 18).

2.1.2 *Web Browser*

Web Browser atau disebut juga dengan *browser*, adalah sebuah *software* yang memungkinkan kita untuk mencari dan mengakses berbagai bagian dari *web*. *Browser* memungkinkan kita menjelajahi web (disebut juga dengan *surf*), seperti mengendarai ombak dengan papan selancar. *Surf* berarti menjelajahi *web* melalui serangkai jalur yang terhubung, atau tautan, dari suatu lokasi, atau dari satu situs *web* ke situs lainnya. (William dan Sawyer, 2010, p. 64).

2.1.3 *Website*

Website, atau biasa disebut hanya dengan *site* (situs), adalah sebuah lokasi pada komputer tertentu di dalam *web* yang memiliki alamat yang unik (disebut dengan *URL).* Sebuah *website* terdiri dari halaman *web* (*web pages*) atau kumpulan dari halaman *web* lainnya yang terkait. (William dan Sawyer, 2010, p. 65).

2.1.4 *Web Pages*

Web Pages atau halaman *web* adalah sebuah dokumen yang ada di dalam *World Wide Web* yang mencakup teks, gambar, suara, dan video. (William dan Sawyer, 2010, p. 65).

2.1.5 *Hypertext Transfer Protocol* (HTTP)

Hypertext Transfer Protocol atau biasa disingkat menjadi HTTP merupakan sebuah protokol *web* yang dikembangkan oleh Tim Berners-Lee. HTTP muncul di bagian awal dari alamat *web*. HTTP sendiri memiliki definisi yaitu aturan-aturan komunikasi yang memungkinkan *browser* untuk terhubung dengan *web server*. (William dan Sawyer, 2010, p. 66).

2.1.6 *Uniform Resource Locator* (URL)

Uniform Resource Locator atau disebut juga dengan URL adalah serangkaian karakter yang mengarah kepada sebuah potongan informasi yang spesifik di manapun di dalam *web*. (William dan Sawyer, 2010, p. 65).

URL terdiri dari (1) protokol *web,* (2) nama *domain* atau nama server *web,* (3) direktori (atau *folder*) di dalam server tersebut, dan (4) *file* di dalam direktori tersebut (dapat juga dengan perpanjangan seperti *html* atau *htm*). Berikut ini adalah contoh dari URL dari situs Kopigenik, yaitu http://www.kopigenik.com/wordpress/index.html

1. *Protocol* : http://

   Protocol adalah sebuah perangkat aturan komunikasi yang mengatur pertukaran informasi. Protokol *web,* yaitu HTTP (*Hypertext Transfer Protocol*), dikembangkan oleh Tim Berners-Lee. HTTP ini dapat kita temukan di bagian awal dari setiap alamat *web*. (William dan Sawyer, 2010, p. 66).

2. *Domain name (web server name)* : www.kopigenik.com/

   Domain adalah sebutan untuk sebuah lokasi di dalam Internet, atau di dalam suatu *web server*. Nama *domain* memberitahukan lokasi dan tipe dari suatu alamat *web*. Komponen dari nama *domain* dipisahkan oleh titik, atau biasa disebut "*dots*". Bagian akhir dari *domain* disebut dengan *top-level domain*. *Top-level domain* adalah ekstensi yang terdiri dari tiga huruf, yang menjelaskan tipe dari *domain* tersebut. Misal, *.gov* (pemerintah), *.com* (komersial), *.net* (jaringan), *.edu* (pendidikan), *.org* (organisasi/nirlaba), *.mil* (militer), dan *.int* (organisasi internasional). (William dan Sawyer, 2010, p. 66).

3. *Directory name* : wordpress/

   Directory name atau disebut juga nama direktori adalah nama yang terdapat pada server untuk sebuah direktori, atau *folder*, yang dibutuhkan oleh browser untuk sebagai destinasi untuk lokasi pengambilan *file* dan informasi lainnya. (William dan Sawyer, 2010, p. 67).

4. *File name* dan *extension* : index.html

   File adalah sebuah halaman atau dokumen tertentu yang kita cari. Misal, pada contoh URL terdapat *index.html*. Hal tersebut menunjukkan/menampilkan tampilan *file* dari *home page* atau halaman awal. *.html* merupakan ekstensi dari *file* yang berisi halaman awal. Ekstensi tersebut menginformasikan kepada browser bahwa *file* tersebut merupakan *file* HTML.

2.2 Interaksi Manusia Komputer (IMK)

Menurut Shneiderman dan Plaisant (2010, p. 4), Interaksi Manusia dan Komputer adalah ilmu desain yang merupakan gabungan dari pengumpulan data dan penggunaan kerangka kerja menggunakan alat-alat yang dikembangkan dari ilmu komputer.

2.2.1 *Eight Golden Rules*

Di dalam membuat suatu rancangan antarmuka untuk suatu aplikasi, Shneiderman dan Plaisant (2010, pp. 88-89) mengusulkan 8 *Golden Rules* :

1.*Strive for consistency*.

Konsistensi sangat penting di dalam perancangan antarmuka, seperti langkah-langkah yang konsisten untuk situasi yang sama, istilah yang serupa untuk menu serta warna dan tulisan yang konsisten.

2. *Cater to universal usability*.

Desain antarmuka harus dapat digunakan oleh untuk berbagai macam pengguna. Perbedaan usia, kecacatan, hingga pengalaman di dalam menggunakan aplikasi akan mendasari pembuatan desain. Dengan memberikan penjelasan kepada para pemula, dan jalan pintas (*shortcut*) untuk yang sudah biasa menggunakan aplikasi, dapat membantu meningkatkan pengertian terhadap sistem.

3. *Offer informative feedback.*

Untuk setiap hal yang dilakukan *user* harus adanya umpan balik dari sistem. Sistem akan bereaksi berdasarkan aksi yang dilakukan *user*, baik aksi tambahan maupun aksi utama.

4. *Design dialogs to yield closure*.

Urutan aksi harus dipisahkan menjadi awal, tengah, dan akhir. Umpan balik yang diberikan setiap kali pengguna menyelesaikan suatu aksi, akan memberikan informasi, rasa puas, serta merupakan tanda untuk mempersiapkan aksi selanjutnya.

5. *Prevent errors*.

Sebisa mungkin rancanglah sistem yang penggunanya tidak dapat membuat kesalahan yang serius, dan meskipun terjadi kesalahan, sistem harus dapat membacanya dan memberikan solusi yang jelas dan sederhana. Non-aktifkan menu yang tidak perlu dan lakukan validasi pada bagian input.

6. *Permit easy reversal of actions.*

Buatlah fitur "kembali" (*reverse/undo*) untuk sebanyak mungkin aksi. Fitur ini memberikan rasa aman karena pengguna tahu bahwa aksi yang belum tepat dapat diulang kembali.

7. *Support internal locus of control.*

Mempermudah para pengguna yang sudah terbiasa menggunakan sistem. Mereka tidak menyukai perubahan-perubahan pada sistem yang menyebabkan sistem tidak berjalan sesuai dengan apa yang mereka harapkan, dan akhirnya mereka kesulitan di dalam mendapatkan informasi tertentu.

8. *Reduce short-term memory load.*

Keterbatasan ingatan manusia mengharuskan para perancang untuk membuat antarmuka yang tidak membuat penggunanya untuk harus mengingat banyak hal untuk digunakan di halaman-halaman selanjutnya.

2.3 E-Business

E-Business adalah semua bentuk pertukaran informasi yang dimediasi secara elektronik antara sebuah organisasi dan *stakeholder* eksternalnya, yang mendukung jangkauan proses bisnis.

2.3.1 E-Commerce

E-Commerce adalah semua bentuk pertukaran informasi yang dimediasi secara elektronik antara sebuah organisasi dan *stakeholder* eksternalnya. *E-commerce* seringkali dianggap hanya sebagai transaksi jual beli menggunakan Internet. Ketika orang mendengar kata *e-commerce*, orang langsung mengacu kepada penjualan ritel dari perusahaan, contohnya seperti Amazon. Akan tetapi, *e-commerce* tidak hanya sekedar transaksi finansial yang dimediasi secara elektronik antara organisasi dan pelanggan. *E-commerce* lebih mengacu kepada semua transaksi yang dimediasi secara elektronik antara sebuah organisasi dan pihak ketiga yang terlibat.

Dengan kata lain, transaksi non-finansial seperti permintaan pelanggan untuk informasi lebih mengenai suatu produk atau jasa juga dapat disebut sebagai bagian dari *e-commerce*. (Chaffey, 2011, p. 10).

Menurut Chaffey (2011, p.10), *e-commerce* memiliki beberapa perspektif, yaitu :

1. Perspektif komunikasi : penyampaian informasi, produk, atau layanan, atau pembayaran dengan cara elektronik.
2. Perspektif proses bisnis : penerapan teknologi terhadap otomasi transaksi bisnis dan alur kerja.
3. Perspektif layanan : memungkinkan pemotongan biaya, sekaligus meningkatkan kecepatan dan kualitas pelayanan.
4. Perspektif *online* : pembelian dan penjualan produk dan informasi secara *online*.

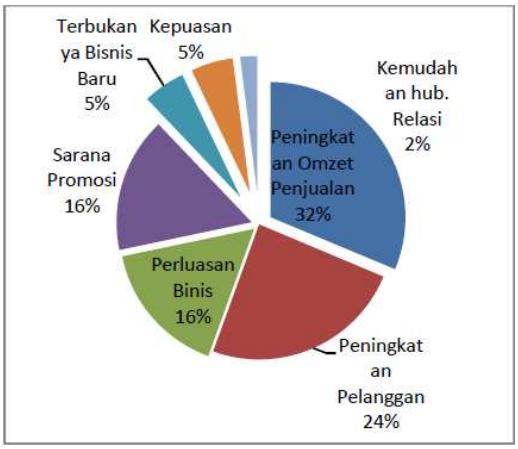
Menurut Chaffey (2011, p.11), berdasarkan transaksi *e-commerce* antar organisasi, *e-commerce* terbagi menjadi dua, yaitu :

1. *Buy-side e-commerce* : mengacu pada transaksi untuk mendapatkan sumber daya yang dibutuhkan oleh organisasi dari *supplier*.
2. *Sell-side e-commerce* : mengacu pada transaksi yang terkait dengan menjual produk ke pelanggan atau organisasi.

2.3.2 Manfaat *E-Commerce*

Menurut Marwati (2013, p. 77), terdapat tujuh manfaat utama dari *e-commerce*, yaitu :

1. Dapat meningkatkan omset penjualan.
2. Dapat meningkatkan jumlah pelanggan.
3. Sebagai perluasan jangkauan bisnis.
4. Sebagai sarana promosi.
5. Dapat membuka peluang untuk bisnis baru.
6. Untuk kepuasan pelanggan.
7. Kemudahan untuk hubungan relasi.



Gambar 2.1 Diagram Manfaat *E-commerce*

Sumber : Marwati, 2013, p. 77.

2.4 Rekayasa Perangkat Lunak (*Software Engineering*)

Scrum adalah salah satu pendekatan *agile* yang digunakan untuk mengembangkan produk dan jasa. Kata *scrum* bukanlah sebuah akronim, melainkan sebuah istilah dari dunia olahraga *rugby*, yang berarti memulai ulang permainan setelah terjadi pelanggaran yang tidak disengaja atau saat bola sudah tidak dapat dipakai. *Scrum* digambarkan sebagai pendekatan yang berbasis tim menuju pengembangan produk yang bersifat semua-sekaligus. (Rubin K., 2013, pp. 1-3).

2.4.1 *Scrum Roles*

Di dalam pengembangan aplikasi menggunakan *scrum*, terdapat 1 atau lebih tim *scrum*. Di dalam tim *scrum* tersebut terdapat 3 *scrum roles* : *product owner*, *ScrumMaster*, dan tim pengembangan (*development team*).

1. *Product Owner*

   Product owner merupakan titik pusat utama dari kepemimpinan pengembangan produk. Dia bertanggung jawab untuk menentukan fitur dan fungsi mana yang harus dibuat, serta urutannya. *Product owner* mengkomunikasikan visi yang jelas mengenai apa yang akan dibuat kepada semua anggota tim. *Product owner* bertanggung jawab atas keseluruhan keberhasilan solusi yang sedang dikembangkan.

Untuk memastikan tim bekerja sesuai dengan apa yang *product owner* inginkan, *product owner* berkolaborasi dengan *ScrumMaster* dan tim pengembangan. (Rubin K., 2013, pp. 15-16).

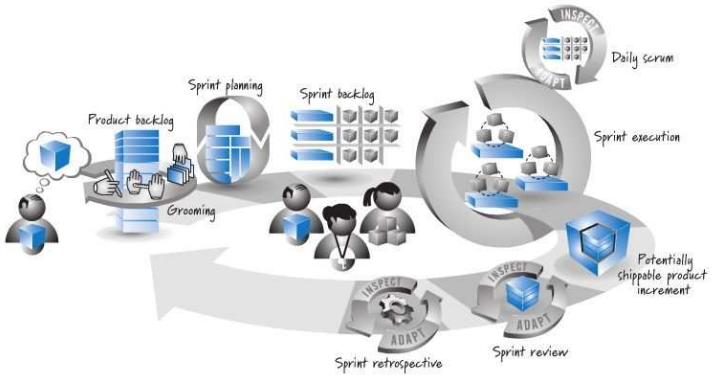
2. *ScrumMaster*

   ScrumMaster membantu setiap orang yang terlibat untuk memahami nilai dan prinsip dari *scrum*. Dia bertindak sebagai *coach*, memimpin, serta membantu tim *scrum* dan seluruh organisasi untuk mengembangkan pendekatan *scrum* yang cocok dan spesifik untuk organisasi mereka. *ScrumMaster* juga membantu organisasi untuk melewati proses adaptasi dalam mengadopsi *scrum*. *ScrumMaster* tidak sama dengan *project manager*, karena *ScrumMaster* tidak memiliki wewenang untuk sepenuhnya mengontrol tim. *ScrumMaster* hanya memimpin dan membantu tim untuk menyelesaikan masalah dan membuat perbaikan sesuai dengan *scrum*. *ScrumMaster* bertindak sebagai pemimpin, bukan *manager*. (Rubin K., 2013, p. 16).

3. Tim pengembangan

   Tim pengembangan menentukan sendiri cara terbaik untuk mencapai tujuan yang sudah ditetapkan oleh *product owner*. (Rubin K., 2013, p. 16).

2.4.2 *Scrum Framework*



Gambar 2.2 *Scrum Framework*

Sumber : Rubin K., 2013, p. 17

1. *Product Backlog*

   Product backlog adalah sebuah daftar yang berisi urutan dari pekerjaan sesuai dengan prioritas yang sudah ditentukan oleh *product owner*. Pada tahap awal dari pengembangan produk baru, *product backlog* berisi fitur-fitur yang dibutuhkan untuk memenuhi visi dari *product owner*. Pada tahap pengembangan produk yang sudah berlangsung, *product backlog* dapat berisi fitur baru, perubahan-perubahan pada fitur yang sudah ada, *defect* yang perlu diperbaiki, perbaikan secara teknis, dll. (Rubin K., 2013, p. 18).

2. *Sprints*

   Dalam *Scrum*, semua pekerjaan dilakukan di dalam iterasi atau siklus yang dapat memiliki durasi hingga 1 bulan yang disebut dengan *sprint.*

   Pekerjaan yang diselesaikan pada setiap *sprint* harus menciptakan sesuatu yang bernilai nyata bagi pelanggan atau pengguna. Salah satu karakteristik dari *sprint* adalah memiliki tanggal mulai dan tanggal selesai. (Rubin K., 2013, p. 20).

3. *Sprint Planning*

   Sprint planning adalah bagian pertama dari setiap *sprint*. *Sprint planning* bertujuan untuk menentukan bagian dari *product backlog* yang paling penting untuk dikerjakan pada *sprint* berikutnya. (Rubin K., 2013, p. 21).

4. *Sprint Execution*

   Sprint execution adalah pekerjaan yang dilakukan oleh tim *Scrum* untuk memenuhi *sprint goal* atau tujuan *sprint*. (Rubin K., 2013, p. 347).

5. *Daily Scrum*

   Setiap hari saat menjalankan *sprint*, tim pengembangan mengadakan sebuah rapat yang biasa disebut *daily scrum* atau disebut juga *daily stand-up* yang berasal dari kebiasaan orang-orang yang berdiri pada saat rapat. Masing-masing anggota tim akan menjawab 3 pertanyaan, yaitu :

- Apa yang sudah saya capai sejak *daily scrum* terakhir ?
- Apa yang saya rencanakan untuk dikerjakan sampai *daily scrum* selanjutnya ?
- Apa hambatan yang saya rasakan yang menghambat saya untuk membuat kemajuan ? (Rubin K., 2013, pp. 23-24).

6. *Potentially Shippable Product Increment*

   Di dalam *scrum*, hasil dari *sprint* disebut dengan *potentially shippable product increment*, yang berarti apapun yang sudah disetujui oleh tim *scrum* sudah benar-benar terpenuhi sesuai dengan definisi terpenuhi yang juga sudah disetujui bersama. (Rubin K. 2013, p. 25).

7. *Sprint Review*

   Di akhir dari sebuah *sprint*, terdapat dua kegiatan tambahan berupa pemeriksaan dan adaptasi. Salah satunya adalah *sprint review*. Tujuan dari kegiatan ini adalah untuk memeriksa dan menyesuaikan produk yang sedang dibangun. Dalam percakapan *sprint review* hal-hal yang diulas hanya fitur-fitur yang sudah selesai dalam konteks keseluruhan dari upaya pengembangan. (Rubin K., 2013, p. 26).

8. *Sprint Retrospective*

   Di akhir dari sebuah *sprint*, terdapat dua kegiatan tambahan berupa pemeriksaan dan adaptasi. Kegiatan kedua dari pemeriksaan dan adaptasi adalah *sprint retrospective*. *Sprint retrospective* adalah kesempatan untuk memeriksa dan mengadaptasi proses dari *sprint*. Dalam percakapan *sprint retrospective* hal-hal yang diulas adalah apa yang bekerja dan apa yang tidak bekerja dengan *scrum*. (Rubin K., 2013, pp. 27-28).

2.5 Database

Database adalah kumpulan dari data yang berhubungan secara logis dan deskripsinya, yang dirancang untuk memenuhi kebutuhan informasi dari sebuah organisasi. *Database* juga dapat diartikan sebagai sebuah repositori data yang

besar, yang dapat digunakan oleh banyak departemen dan pengguna secara bersamaan. *Database* tidak hanya dimiliki oleh satu departemen, tetapi sudah merupakan sumber daya perusahaan yang sifatnya dipakai bersama-sama. *Database* juga tidak hanya menyimpan data operasional dari organisasi saja. *Database* juga menyiman deskripsi dari data itu sendiri. Oleh karena itu, *database* juga didefinisikan sebagai *self-describing.* Data yang bersifat deskriptif disebut dengan *metadata* atau kamus data, yang berarti data tentang data. (Connolly dan Begg, 2015, p. 63).

2.6 *Entity Relationship Diagram* (ERD)

Entity Relationship Diagram atau biasa disebut ERD adalah sebuah diagram yang digunakan pada saat merancang sebuah *database*. ERD dapat menunjukkan informasi-informasi seperti *entity*, *relationship*, *constraint*, dan *attribute.* Tahapan membuat ERD dimulai dengan mengidentifikasi *entity* (entitas) dan *relationship* (hubungan) antar data yang kemudian digambarkan pada sebuah model. (Connolly dan Begg, 2015,405).

1. *Entity*

   Entity type atau biasa disebut *entity* (entitas) adalah sekelompok objek yang memiliki sifat yang sama, yang diidentifikasi oleh perusahaan sebagai objek yang memiliki eksistensi independen. Entitas mewakili sekelompok "objek" di "dunia nyata" dengan sifat yang sama. Entitas memiliki eksistensi independen yang dapat berupa objek yang bersifat fisik (atau nyata, seperti *staff*, properti, dan *customer*) atau objek yang bersifat konseptual (seperti penjualan dan pengalaman kerja). (Connolly dan Begg, 2015, pp. 406-408).

2. *Relationship*

   Relationship type atau biasa disebut *relationship* (hubungan) adalah kumpulan asosiasi antar satu entitas yang berpartisipasi atau lebih. Setiap hubungan memiliki nama yang menggambarkan fungsi dari hubungan tersebut. (Connolly dan Begg, 2015, p. 408).

3. *Attributes*

Attributes atau atribut adalah sifat khusus dari suatu tipe entitas. Atribut menyimpan nilai yang menjelaskan/menggambarkan setiap kejadian/peristiwa dari entitas dan mewakili bagian utama dari data yang tersimpan di dalam *database*. (Connolly dan Begg, 2015, p. 413).

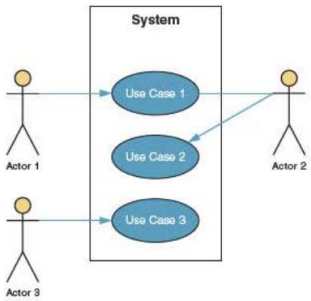
2.7 *Unified Modelling Language* (UML)

Menurut Whitten & Bentley (2007, p. 371), Unified Modelling Language (UML) adalah suatu susunan model yang digunakan untuk menentukan serta menggambarkan suatu sistem dari segi objek-objek.

Menurut Whitten dan Bentley (2007, p. 381) UML memiliki beberapa diagram maupun bagan untuk memberikan sudut pandang yang berbeda terhadap sistem antara lain:

2.7.1 *Use Case Diagram*

Use Case Diagram menggambarkan interaksi antara sistem dengan manusia maupun dengan sistem yang berasal dari luar. *Use Case* menjelaskan siapa yang menjalankan dan bagaimana sistem diharapkan akan berjalan (Whitten and Bentley, 2007, p. 382). Menurut Whitten & Bentley (2007, p. 246) pemodelan *use-case* menjelaskan fungsi dan peran sistem dengan cara menggambarkan sistem dari sudut pandang pengguna dengan cara yang mudah untuk mengerti.

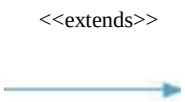
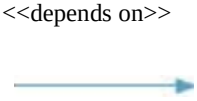


Gambar 2.3 Contoh *Use Case*

Sumber : (Whitten & Bentley, 2007, p. 246)

Use Case memiliki simbol dan hubungan *(Relationship)* sebagai berikut (Whitten & Bentley, 2007, p. 248):

Tabel 2.1 Notasi pada *Use Case*

Nama	Keterangan	Contoh
Actor	Subjek yang akan berinteraksi dengan sistem	
Association	Hubungan yang menggambarkan interaksinya antara keduanya aktor dan *use case*	
Extends	Simbol yang digunakan pada saat ada *use case* yang terdiri dari beberapa langkah, dan untuk menyederhanakannya harus dibuat *use case* yang terpisah	
Uses (or includes)	Adanya use case dengan fungsi yang serupa sehingga perlu dibuat *abstract use case* untuk mengurangi *use case* yang redundan. Hubungan antara *abstract use case* dengan *use case* yang menggunakannya disebut sebagai *uses*.	
Depends on	Simbol yang menjelaskan bahwa suatu *use case* tidak bisa berjalan sebelum *use case* tertentu telah selesai dikerjakan	

Inheritance	Simbol yang dibuat untuk menyederhanakan gambar ketika suatu aktor menurunkan sifat dan peran dari aktor lainnya	

2.7.2 *Activity Diagram*

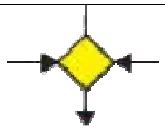
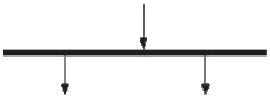
Whitten & Bentley (2007, p. 390) menjelaskan bahwa *activity diagram* adalah diagram yang dapat menggambarkan suatu proses bisnis maupun langkah-langkah dari suatu *use case.* Terdapat beberapa notasi yang digunakan di dalam *Activity Diagram* (Whitten & Bentley, 2007, p. 391) :

Tabel 2.2 Notasi pada *Activity Diagram*

Nama	Keterangan	Contoh
Initial Node	Simbol berbentuk bulat padat yang mewakili mulainya suatu proses	
Actions	Simbol berbentuk kotak dengan sudut yang melengkung, menjelaskan langkah-langkah tertentu pada *activity diagram*	

Flow	tanda panah yang menunjukan arah dari suatu aksi ke aksi lainnya	

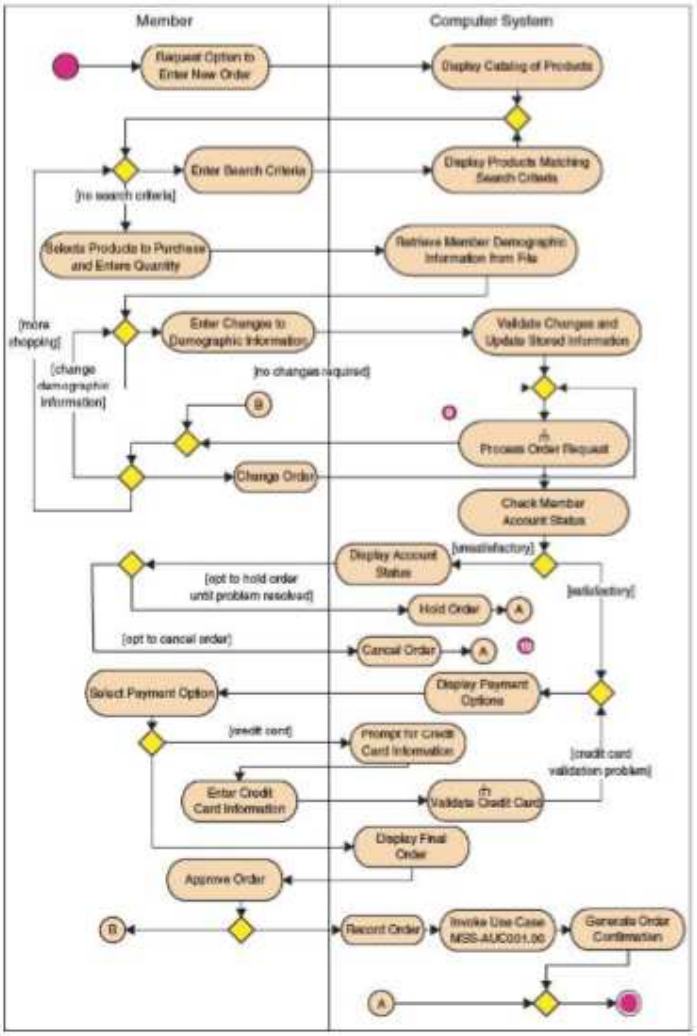
Tabel 2.3 Notasi pada *Activity Diagram* (lanjutan)

Decision	Simbol yang digunakan untuk mewakili beberapa kondisi yang berhubungan dengan aksi yang berbeda	
Merge	Simbol yang menggabungkan alur yang sebelumnya terpisahkan oleh simbol *decision*	
Fork	Simbol yang menjelaskan bahwa aksi yang di bawahnya dapat berjalan secara bersamaan	

Join	Simbol yang menjelaskan bahwa semua alur dan aksi yang menunjuk pada *join* harus diselesaikan terlebih dahulu sebelum lanjut ke proses selanjutnya	

Tabel 2.4 Notasi pada *Activity Diagram* (lanjutan)

Activity Final	Simbol yang mewakili akhir dari proses tersebut	



Gambar 2.4 Contoh *Activity Diagram*

Sumber : (Whitten & Bentley, 2007, p. 393)

2.7.3 *Class Diagram*

Class Diagram adalah gambaran akan struktur objek suatu sistem yang menunjukkan kelas-kelas yang terdapat di dalam sistem serta hubungan antar kelas-kelas tersebut (Whitten & Bentley, 2007, p. 400).

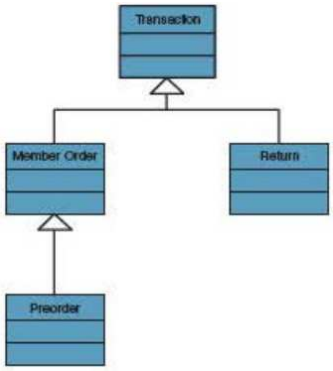
Di dalam diagram ini terdapat *multiplicity (association), generalization/specialization relationship,* dan *aggregation relationship*:

- *Association* adalah hubungan berdasarkan apa yang perlu diketahui oleh salah satu objek tentang objek lainnya. Setelah *association* telah ditentukan maka harus menentukan *multiplicity* yang mempengaruhi asosiasi tersebut.

Gambar 2.5 *Association on Class Diagram*

Sumber : (Whitten & Bentley, 2007, p. 406)

- *Generalization/Specialization Relationship* merupakan hirarki pengelompokkan yang terdiri dari kelas *supertype (abstract or parent)* dan kelas *subtype (concrete or child).*



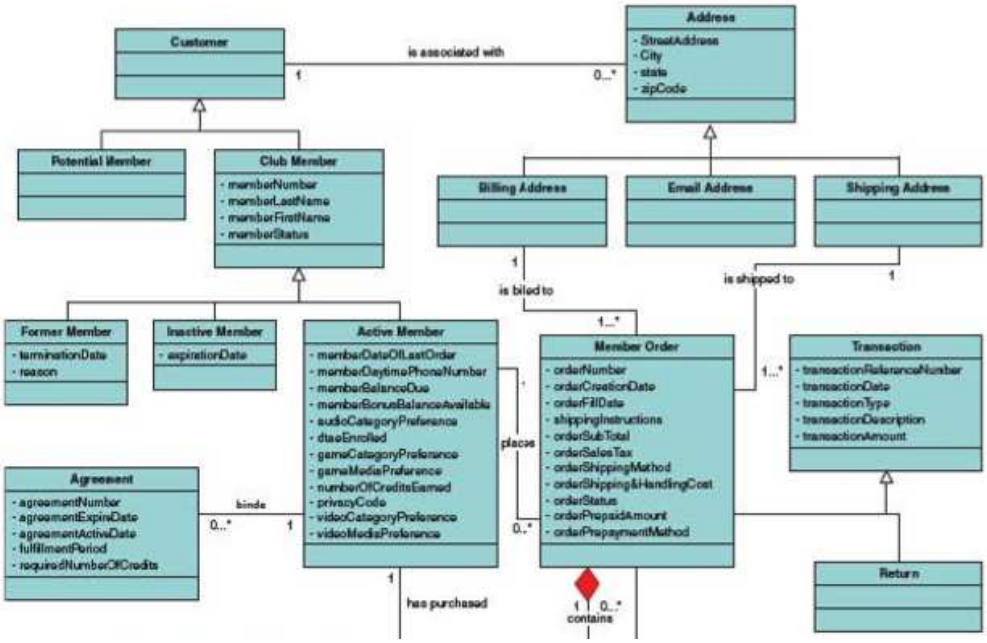
Gambar 2.5 *Generalization in Class Diagram*

Sumber : (Whitten & Bentley, 2007, p. 404)

- *Aggregation/Composition* merupakan tipe keunikan dari suatu hubungan dimana suatu objek merupakan bagian dari objek lainnya.

Gambar 2.6 ***Aggregation Symbol in Class Diagram***

Sumber : (Whitten & Bentley, 2007, p. 406)



Gambar 2.7 Contoh ***Class Diagram***

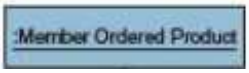
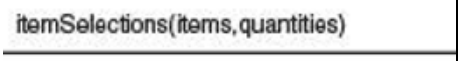
Sumber : (Whitten & Bentley, 2007, p. 406)

2.7.4 *Sequence Diagram*

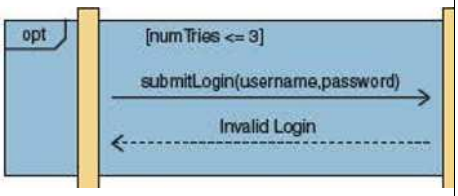
Sequence Diagram adalah diagram yang dibuat berdasarkan suatu use case dan menggambarkan interaksi antar objek dalam urutan waktu (Whitten & Bentley, 2007, p. 659). Tabel 2.3 menjelaskan contoh penggunaan notasi *sequence diagram* menurut Whitten & Bentley (2007, p. 660) :

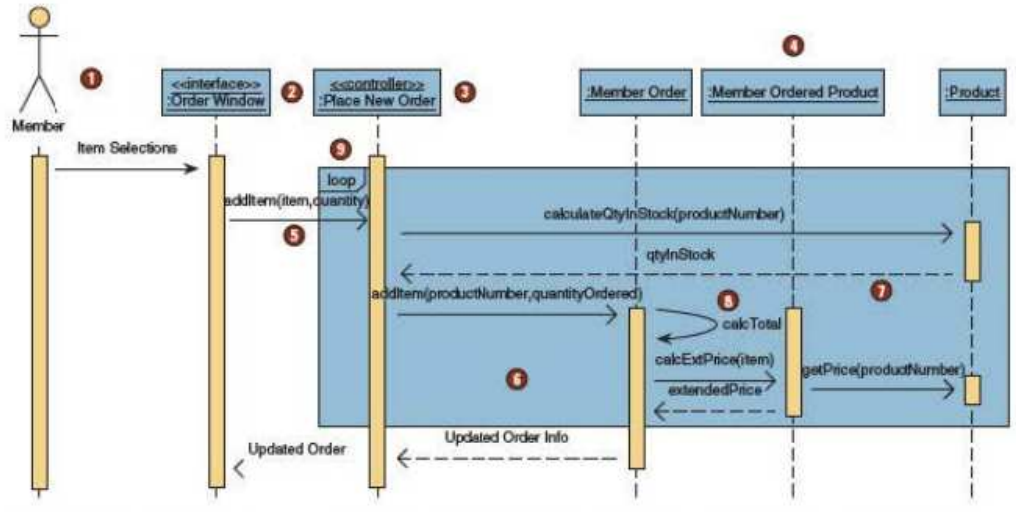
Tabel 2.3 Notasi pada ***Sequence Diagram***

Nama	Keterangan	Contoh

Actor	Aktor yang berinteraksi dengan *interface* pengguna	
Interface class	Kotak yang menandakan kelas *interface* pengguna	
Controller class	Kotak yang menandakan kelas *controller*. Setiap *use case* memiliki satu atau lebih *controller class*	
Entity classes	Benda atau kelas lain yang diperlukan di dalam proses	
Messages	Tanda panah yang menandakan input yang dikirimkan ke kelas	
Activation bars	Berbentuk batang yang mengindikasikan waktu pada saat suatu objek muncul	
Return messages	Tanda panah putus-putus yang menunjukkan pesan yang mengembalikan suatu nilai	

Tabel 2.3 Notasi pada *Sequence Diagram* (lanjutan)

Frame	Simbol berbentuk bingkai yang mengelompokkan suatu *sequence* tertentu yang merupakan pengulangan (*loop*) ataupun *optional*	



Gambar 2.8 Contoh *Sequence Diagram*

Sumber : (Whitten & Bentley, 2007, p. 659)

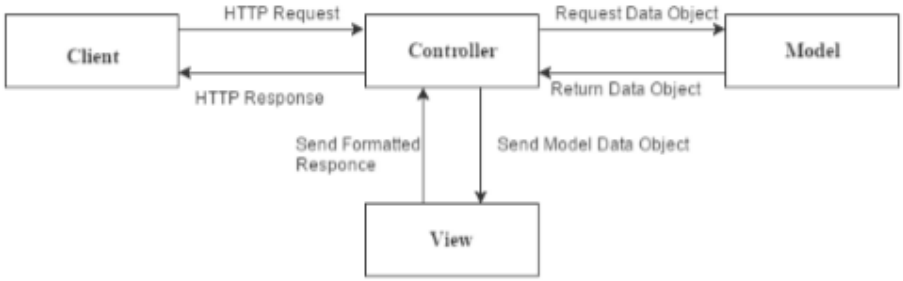
2.8 *Hypertext Preprocessor* (PHP)

PHP adalah suatu bahasa pemrograman yang digunakan pada sisi *server* untuk menghasilkan *output* yang dinamis. Ini artinya output yang berbeda dapat dihasilkan setiap kali *browser* melakukan *request* terhadap halaman tertentu (Nixon, 2014, p. 35). Di dalam menghasilkan *web* yang dinamis, PHP dapat dikombinasikan dengan MySQL, JavaScript, CSS, dan HTML5.

PHP dapat digunakan secara gratis dan merupakan bahasa pemrograman *open-source* yang artinya PHP telah dikembangkan oleh berbagai *programmer* berdasarkan apa yang mereka perlukan dan inginkan. Peran PHP adalah untuk menangani bagian *web server*, sedangkan MySQL akan menangani semua data yang diperlukan oleh *client*, dan HTML5, CSS, serta JavaScript berperan di dalam penyajian tampilan halaman *web* (Nixon, 2014, p. 12).

2.9 *Model View Controller* (MVC)

Model View Controller (MVC) adalah pola yang paling banyak digunakan pada pembuatan aplikasi *web*. *Model* menangani data-data yang digunakan oleh aplikasi, dengan cara menyimpan maupun mengambil data dari *database*, serta menyimpan fungsi-fungsi yang digunakan oleh aplikasi. *View* bertanggung jawab di dalam mengambil dan menampilkan data yang didapatkan dari *model*. *Controller* berfungsi sebagai penengah di dalam mengatur kerjasama antara *view* dan *model*. *Controller* menerima permintaan dari pengguna dan meminta *model* untuk menjalankan permintaan tersebut dan juga mengirimkan data yang dibutuhkan, dan akhirnya mengirimkan informasi tersebut kepada *view* untuk ditampilkan kepada *client* (Parkar, Shinde, Gadade, dan Shinde, 2016, p. 38).



Gambar 2.9 Diagram Kolaborasi MVC

Sumber : (Parkar, Shinde, Gadade, dan Shinde,2016, p. 39)

2.9.1 Laravel

Laravel merupakan salah satu *framework* MVC yang menggunakan bahasa pemrograman PHP. Laravel memiliki beberapa *tools* yang digunakan seperti *database migrations* dan *Eloquent* yang memungkinkan penggunanya untuk memanipulasi data (*create, retrieve, update, delete*) tanpa menuliskan sintaks SQL. Laravel menyederhanakan tugas maupun fungsi yang sering diperlukan di dalam suatu aplikasi seperti otentikasi, *routing, session,* dan *caching* (Parkar, Shinde, Gadade, dan Shinde,2016, p. 39).

Laravel menyediakan fungsi-fungsi sebagai berikut *caching* (Parkar, Shinde, Gadade, dan Shinde,2016, p. 39):

- Keamanan

  Laravel menggunakan *Eloquent* yang mencegah aplikasi dari bahaya seperti *sql-injection.*
- Menyimpan *password*

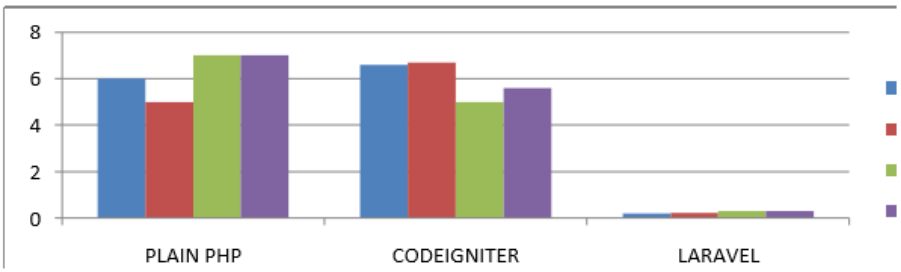
  Laravel menggunakan *BCrypt Hashing* di dalam penyimpanan *password.*
- Pengingat dan mengulang *password*

  Sebagian besar aplikasi *web* menyediakan fitur pengingat dan *reset password* untuk para penggunanya. Laravel menyederhanakan kerja para *developer* di dalam mengembangkan fitur ini di dalam aplikasi mereka.
- Enkripsi

  Laravel memiliki ekstensi *mcrypt* PHP untuk enkripsi AES (*Advanced Encryption Standard*) yang lebih kuat.
- Validasi

  Laravel memiliki kelas *validation* yang dapat dengan mudah digunakan untuk melakukan validasi terhadap i*nput* yang diberikan serta mengembalikan pesan *error* kembali kepada pengguna.

Das dan Saikia (2016, p. 46) di dalam penelitiannya membuktikan bahwa Laravel dapat bekerja lebih cepat di dalam menjalankan *create, read, update,* dan *delete* dibandingkan dengan PHP biasa dan *CodeIgniter* yang juga merupakan salah satu *framework MVC*.



Gambar 2.10 Grafik Perbandingan Waktu Eksekusi

Sumber : (Das & Saikia, 2016, p. 46)

2.10 XAMPP

XAMPP merupakan singkatan dari X (berarti *cross-platform*), Apache, MySQL, dan PHP. XAMPP adalah *web server* yang merupakan kompilasi dari beberapa program yaitu Apache, MySQL dengan menggunakan bahasa pemrograman PHP dan Perl.

XAMPP mudah untuk digunakan di dalam melayani tampilan *web* yang dinamis. Salah satu keuntungan di dalam menggunakan XAMPP yaitu XAMPP mendukung banyak sistem operasi, serta gratis untuk digunakan (Priyanti & Iriani, 2013, p. 56).

2.11 JavaScript

JavaScript merupakan bahasa pemrograman yang diciptakan untuk mengakses seluruh elemen pada dokumen HTML dengan tujuan untuk memanipulasi serta mengontrol elemen-elemen tersebut secara dinamis.

Dengan kata lain JavaScript menyediakan fungsi untuk berinteraksi denga pengguna secara dinamis seperti validasi *input* serta menampilkan suatu pesan tertentu ketika suatu aksi telah dilakukan oleh *user* (Nixon, 2015, p. 9). Berikut adalah contoh dasar penggunaan JavaScript:

```
<script type="text/javascript">
  document.write("Today is " + Date() );
</script>
```

Gambar 2.11 Contoh Penggunaan JavaScript

Sumber : (Nixon, 2015, p. 8)

2.11.1 JQuery

JQuery merupakan *library* atau *framework* dari bahasa pemrograman JavaScript yang berisi fungsi-fungsi yang dibuat untuk menyederhanakan penulisan aplikasi JavaScript. JQuery menyediakan fungsi untuk melakukan operasi-operasi pada elemn-elemen di dalam dokumen (Cameron, 2013, p. 28). JQuery banyak digunakan karena beberapa alasan berikut :

- Mempermudah di dalam menangani masalah yang sering terjadi karena perbedaan *browser* yang digunakan oleh para *user.*
- Menyediakan sangat banyak sintaks yang singkat serta mudah untuk digunakan.
- Lebih mudah membuat *custom plugin* untuk JQuery sesuai dengan kebutuhan.
- Ada sangat banyak *plugin open source* yang tersedia untuk JQuery seperti JQuery UI.

2.12 *Hypertext Markup Language* (HTML), *Cascading Style Sheets* (CSS), Bootstrap

HTML (*Hypertext markup language*) adalah sekumpulan instruksi khusus (biasa disebut "*tags*" atau "*markups*") yang digunakan untuk menekankan struktur dari dokumen, format, dan terhubung pada dokumen multimedia lainnya di dalam *web*. XHTML (*Extensible hypertext markup language*) adalah penerus dan juga versi HTML pada saat ini. Kebutuhan akan versi HTML yang lebih ketat disebabkan oleh konten *World Wide Web* (WWW) pada saat ini yang perlu dikirim ke banyak perangkat (contohnya ponsel) yang memiliki sumber daya lebih sedikit daripada komputer tradisional. (William dan Sawyer, 2010, p. 68).

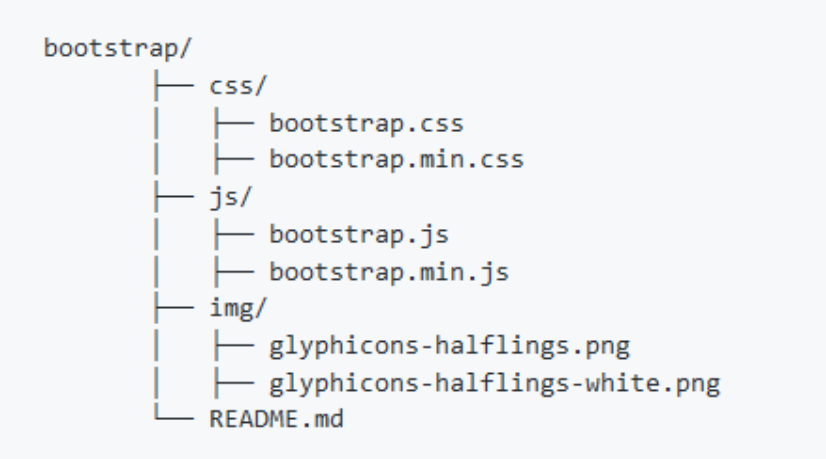
2.13 *Cascading Style Sheets* (CSS)

CSS (*Cascading style sheets*) adalah sebuah *file* yang berisi teks yang menggunakan properti dan nilai untuk menentukan bagaimana elemen tertentu dalam suatu halaman *web* harus ditampilkan. CSS memiliki beragam macam properti, statis dan dinamis, yang memiliki fungsi yang berbeda-beda, seperti mengatur ukuran font dan warna tulisan, tata letak dan posisi elemen, menentukan *page break*, dan juga properti dinamis seperti menu *drop-down* dan komponen interaktif lainnya. (Castro E., 2008, p. 119).

2.14 Bootstrap

Bootstrap adalah sebuah *front-end framework* yang membantu *developer* untuk memulai proses pengembangan web, terutama pada bagian *front-end* atau antarmuka. Bootstrap juga mempermudah *developer* untuk membuat *layout* halaman *web* yang sudah bersifat responsif dengan menggunakan *default grid system*.

Bootstrap juga memiliki *plugin* untuk JavaScript dan *icons*, yang berjalan beriringan dengan *buttos* dan *forms*, sehingga kita dapat menggunakan CSS dan JavaScript yang ada di dalam bootstrap sesuai dengan kebutuhan. (Spurlock J., 2013, p. 1).



Gambar 2.12 Struktur *file* Bootstrap

Sumber : (Spurlock J., 2013, p. 2)

2.15 *Content Management System* (CMS)

Content Management System (CMS) adalah suatu sistem yang berbentuk *interface* yang dapat digunakan para penggunanya untuk menambahkan, menghapus, mengubah, maupun menyimpan konten dari suatu *website*. Hal-hal tersebut dapat dilakukan tanpa harus mempelajari bahasa pemrograman, meskipun CMS harus dipasangkan terlebih dahulu oleh *programmer* sebelum dapat dipakai. CMS memungkinkan beberapa orang untuk berkolaborasi dan sama-sama berkontribusi di dalam menejemen konten *website* (Sharma & Kurhekar, 2013, p. 258). Keuntungan di dalam menggunakan CMS yaitu (Sharma & Kurhekar, 2013, p. 260):

- *Content Management System* hampir selalu dibuat dengan baik oleh para *vendor* dan banyak digunakan.
- Sistem CMS sangat mudah untuk dipasang dan kemudian dijalankan.
- Sangat banyak CMS yang tersedia dan sistem ini membutuhkan biaya yang murah untuk menjalankan sebuah *website*
- Ada komunitas *developer* yang sangat memperhatikan CMS, sehingga ketika muncul suatu masalah dapat segera diselesaikan dengan cepat.
- CMS dapat diimplementasikan dengan hampir semua *database* dan dapat secara mudah bermigrasi ke server yang berbeda.

2.16 *Black-box Testing*

Black-box testing adalah sebuah metode yang digunakan dalam tahap *testing* suatu aplikasi yang menguji fungsionalitas dari aplikasi berdasarkan spesifikasi dari aplikasi tersebut. *Black-box testing* dapat digunakan dalam setiap tingkatan pengujian aplikasi, baik dari pengujian unit, integrasi, sistem, hingga penerimaan (*acceptance*). (tutorialspoint.com, 2017)

2.17 Keragaan Kopi Domestik di Indonesia

Indonesia menduduki peringkat keempat berdasarkan tingkat produksi kopi. Bahkan pada periode sebelumnya, Indonesia pernah menduduki peringkat ketiga setelah Vietnam. Sejalan dengan tingkat produksinya, Indonesia menduduki peringkat keempat berdasarkan tingkat ekspor kopi di pasar dunia. Hal ini menjadikan kopi sebagai salah satu komoditi prioritas negara Indonesia. Namun, jika dibandingkan dengan negara pesaing terdekat seperti Vietnam dan Kolombia, Indonesia memiliki luas lahan untuk kopi yang jauh lebih besar.

Menurut Sagita T. dan Hidayati D. R. (2013, p. 49), keragaan kopi domestik di Indonesia dipengaruhi oleh beberapa faktor seperti produksi, permintaan, dan juga harga kopi di Indonesia. Jumlah produksi kopi dipengaruhi oleh harga kopi Indonesia dan harga kopi pada tahun sebelumnya. Jumlah perminta kopi dipengaruhi oleh pendapatan penduduk dan jumlah penduduk. Sedangkan faktor yang mempengaruhi harga kopi di Indonesia adalah permintaan kopi di Indonesia, konsumsi kopi di Indonesia, dan harga kopi dunia.

2.18 Hasil Penelitian Aplikasi Sejenis

Kopigenik menerapkan prinsip *e-commerce* di dalam memasarkan produk maupun layanan yang disediakan. Menurut Niranjanamurthy, Kavyashee, Jagannath, dan Chahar (2013, p. 2360) e-commerce seringkali diartikan sebagai kegiatan menjual dan membeli suatu produk melalui internet, namun segala transaksi yang sepenuhnya menggunakan teknologi electronic dapat diartikan sebagai e-commerce. Menurut Fatima (2014, p. 71) pada tahun 2013 pasar *e-commerce* di India sudah mencapai 75.000 crore, dan salah satu *e-commerce* terbesar di India, Flipkart, telah mencapai penjualan per tahun sebesar 6.100 crore.

Vidya shree, Bhandari, Sharma, Verma, dan Chauhan (2015, p. 1022) menyimpulkan bahwa 92% orang yang berbelanja secara *online* puas dengan produk maupun jasa yang ditawarkan, 8% tidak puas dikarenakan ketakutan mereka bahwa produk yang ditunjukkan berbeda dengan yang akan mereka dapatkan, dan 88% lebih memilih untuk berbelanja secara *online* dikarenakan kemudahannya, sedangkan 12% orang masih terbiasa untuk berbelanja secara tradisional, beberapa dari mereka khawatir akan mendapatkan produk bekas pakai. Dari fakta-fakta tersebut, dapat disimpulkan bahwa penjualan dengan model bisnis *e-commerce* memiliki potensi penjualan per tahun yang cukup besar. Model bisnis *e-commerce* juga diminati oleh banyak orang, hal tersebut dapat dilihat dari tingkat kepuasan konsumen yang berbelanja secara *online*, serta alasan mereka memilih untuk berbelanja *online*.