BAB II
LANDASAN TEORI

2.1. Tinjauan Pustaka

Beberapa penelitian yang berkaitan dengan penerapan aplikasi *e-commarce* berdasarkan jurnal penelitian terlihat pada Tabel 2.1:

Tabel 2.1 Tinjauan Pustaka

1	Nama (Tahun)	(Armanda and Putra, 2020)
	Judul	Rancang Bangun Aplikasi E-Commerce Untuk Usaha Penjualan Helm
	Masalah	Toko Edi Helm Bandar Lampung yang masih menggunakan sistem konvensional, dimana mengharuskan konsumen datang langsung ke toko untuk melihat, memilih dan membeli helm yang diinginkan. Mengakibatkan informasi produk yang di jual maupun informasi toko itu sendiri masih kurang efektif dengan jangkauan promosi yang tidak luas.
	Metode	*Extreme Programming*
	Hasil	Hasil dari penelitian ini akan menghasilkan sebuah website e-commerce yang akan membantu Toko Edi Helm Bandar Lampung dalam melakukan penjualan secara online sehingga dapat menjangkau lebih banyak minat konsumen dan memperoleh pasar yang besar
2	Nama (Tahun)	(Amirussalam, Putra and Purnomo, 2022)
	Judul	Pengembangan Aplikasi e-Commerce Penjualan Tanaman Anggrek isitaman.com menggunakan Restful API
	Masalah	Masih dilakukan secara manual
	Metode	*Agile Scrum*
	Hasil	Penelitian ini dapat memberikan kontribusi untuk penelitian selanjutnya terkait Restful API dan E-commerce
3	Nama (Tahun)	(Andrianto and Munandar, 2022)
	Judul	Aplikasi E-Commerce Penjualan Pakaian Berbasis Android Menggunakan Firebase Realtime Database
	Masalah	Saat ini masih banyak bisnis jual beli pakaian yang masih menggunakan cara manual dalam proses jual beli yaitu data transaksi jual beli belum mengikuti proses transaksi selayaknya e-commerce yang berkembang pada saat ini seperti pembayaran dan pemilihan produk masih manual sehingga konsumen dalam melakukan transaksi tidak diberikan pelayanan yang baik oleh pihak penjual.

	Metode	*Watrefall*
	Hasil	Aplikasi e-commerce penjualan pakaian berbasis android menggunakan firebase realtime database telah mampu membantu konsumen dalam proses transaksi jual beli pakaian dan konsumen lebih mudah dalam mengetahui informasi mengenai data pembelian yang telah dilakukan
4	Nama (Tahun)	(Isa, 2021)
	Judul	Perancangan aplikasi e-commerce penjualan kayu pada pt. sekar gayanti utama berbasis web
	Masalah	Proses penjualan dan pembelian belum berbentuk website atau online, sehingga menghambat waktu untuk proses selanjutnya seperti penebangan kayu dan pengiriman kayu. Pengembangan
	Metode	*Waterfall*
	Hasil	Hasil akhir dari penelitian ini adalah perancangan aplikasi penjualan kayu berbasis web
5	Nama (Tahun)	(Herfandi, Syahwatullah and Julkarnain, 2023)
	Judul	Rancang Bangun Aplikasi E-Commerce Berbasis Web Pada Toko UD. Nira Utama
	Masalah	Proses penjualan masih dilakukan secara manual
	Metode	*Waterfall*
	Hasil	Hasil Penelitian telah diterapkan untuk memecahkan permasalahan pada Toko UD. Nira Utama
6	Nama (Tahun)	(Zuhri, Muhtadi and Junaedi, 2019)
	Judul	Implementasi Metode Prototype dalam Membangun Sistem Informasi Penjualan Online pada Toko Herbal Pahlawan
	Masalah	Belum adanya sistem secara digital
	Metode	*Prototype*
	Hasil	membangun sistem informasi penjualan secara online pada Toko Herbal Pahlawan supaya memperluas jangkuan pemasaran produknya.

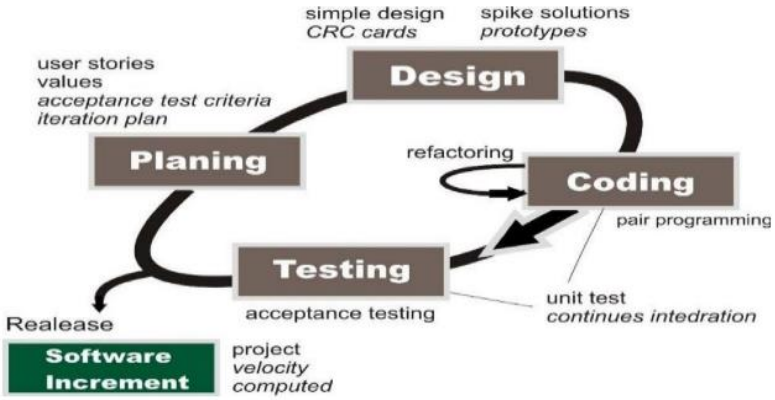
2.2. *E-commerce*

E-commerce merupakan sebuah proses pembelian dan penjualan secara elektronik atas barang atau jasa dan informasi. Secara garis besar, perdagangan elektronik (*e- commerce*) didefinisikan sebagai cara untuk menjual dan membeli ba-rang-barang (dan jasa) lewat jaringan internet (Andrianto and Munandar, 2022)

E-Commerce merupakan proses pembelian dan penjualan jasa atau produk antara dua belah pihak melalui internet (*commerce net*) dan sejenis mekanisme bisnis elektronik dengan focus pada transaksi bisnis berbasis individu dengan menggunakan internet sebagai media pertukaran barang atau jasa baik antar intansi atau individu dengan intansi (*Net-Ready*) (Hendriyati and Yusta, 2021)

2.3. Metode Pengembangan Sistem *Extreme Programming*

Extreme Programming (XP) adalah metodologi pengembangan perangkat lunak yang ditujukan untuk meningkatkan kualitas perangkat lunak dan tanggap terhadap perubahan kebutuhan pelanggan. Jenis pengembangan perangkat lunak yang dimaksudkan untuk meningkatkan produktivitas dan memperkenalkan pos pemeriksaan dimana persyaratan pelanggan baru dapat diadopsi. Tahapan - tahapan dari *Extreme Programming* terdiri dari planning seperti memahami kriteria pengguna dan perencanaan pengembangan, designing seperti perancangan prototype dan tampilan, coding termasuk pengintegrasian, dan yang terakhir adalah testing (Ependi, Usman, and Widayati 2013). Model *Extreme Programming* dapat dilihat pada Gambar 2.1.



Gambar 2. 1 Tahapan pada Extreme Programming (XP).
Sumber : (Ependi, Usman, and Widayati 2013)

Dibawah 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 integration plan* melakukan perhitungan kecepatan *project* selama *development, customer* dapat menambah *story*, merubah *value*, membagi story atau menghapusnya.

2. *Design* (Perancangan)

Perancangan yang simple, menarik, dan sederhana selalu memberikan hasil disukai daripada gambaran-gambaran yang lebih kompleks. Perancangan XP memberikan panduan implementasi untuk suatu cerita ketika ditulis, tidak kurang, tidak lebih.Terdapat simple design CRC Cards 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 16 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 kode dari satu story. Dan

refactoring adalah proses restrukturisasi kode program komputer yang ada tanpa mengubah perilaku eksternalnya.

4. *Testing* (Pengujian)

Unit pengujian yang harus dibuat dan kemudian dijalankan menggunakan kerangka kerja yang memungkinkan mereka untuk diotomatisasi sehingga dapat dijalankan dengan mudah dan dapat dijalankan berulang kali. Pada tahapan pengujian yaitu unit *test continuous 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 release. Selanjutnya terdapat tahapan *software increment project velocity computed* yaitu tahapan yang telah diimplementasikan dari *software release* yang nantinya akan diterapkan dalam suatu sistem.

2.4. Alat Implementasi

2.4.1. Xampp

Menurut MADCOMS (2016) Xampp adalah sebuah paket kumpulan software yang terdiri dari Apache, MySQL, PhpMyAdmin, PHP, Perl, Filezilla, dan lain.

Xampp berfungsi untuk memudahkan instalasi lingkungan PHP, di mana biasanya lingkungan pengembangan web memerlukan PHP, Apache, MySQL dan PhpMyAdmin (MADCOM, 2016).

2.4.2. *Dreamwever*

Adobe Dreamweaver adalah :aplikasi desain dan pengembangan web yang menyediakan editor WYSIWYG visual (bahasa sehari-hari yang disebut sebagai Design view) dan kode editor dengan fitur standar seperti syntax highlighting, code completion, dan code collapsing serta fitur lebih canggih seperti real-time syntax checking dan code introspection untuk menghasilkan petunjuk kode untuk membantu pengguna dalam menulis kode (Destiningrum and Adrian, 2017).

2.4.3. *MySQL*

Menurut MADCOM (2016) MySQL adalah sistem manajemen Database SQL yang bersifat Open Source dan paling popular saat ini. Sistem Database MySQL mendukung beberapa fitur seperti multithreaded, multiuser dan SQL Database managemen system (DBMS).

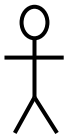
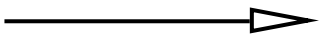
2.5. Bahasa Pemodelan Pengembangan Sistem (*UML*)

Bahasa Pemodelan Pengembangan Sistem (*Unified Modeling Language*) adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement* (kebutuhan), membuat analisis & desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek (Rosa & Shalahuddin, 2018).

2.5.1. *Use Case Diagram*

Use Case Diagram atau diagram *use case* merupakan pemodelan untuk kelakuan *(behavior)* sistem informasi yang akan dibuat (Rosa & Shalahuddin, 2018). *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat menjelaskan simbol-simbol yang ada pada diagram *use case* dapat dilihat pada gambar 2.2 di bawah ini:

Tabel 2. 2 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, biasanya dinyatakan menggunakan kata benda di awal frase nama *actor*
Asosiasi/*association* 	Komunikasi antara aktor dan *use case* yang berpatisipasi pada *use case* atau *use case* memiliki interaksi dengan *actor*
Ekstensi/*extend*	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan
<<*extend*>> 	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*	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan

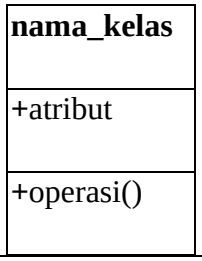
<<*include*>> 	memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini

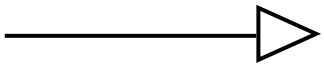
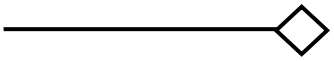
Sumber : (Rosa & Shalahuddin, 2018)

2.5.2. *Class Diagram*

Diagram kelas atau *Class Diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi (Rosa and Shalahudin, 2018). menjelaskan simbol-simbol yang ada pada diagram kelas pada tabel *class diagram* 2.3.

Tabel 2. 3 Simbol *Class Diagram*

Simbol	**Deskripsi**
Kelas  	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*

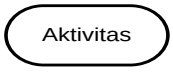
Generalisasi 	Relasi antar kelas dengan makna generalisasi-spesialisasi(umum khusus)
Kebergantungan/*dependecy* 	Relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi/*agregation* 	Relasi antar kelas dengan makna semua bagian (*whole-part*)

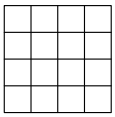
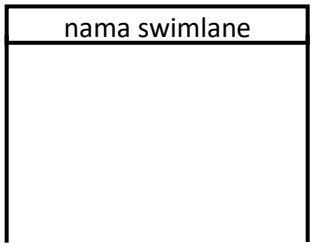
Sumber : (Rosa & Shalahuddin, 2018)

2.5.3. *Activity Diagram*

Activity Diagram atau Diagram aktivitas menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis (Rosa & Shalahuddin, 2018), menjelaskan Simbol-simbol yang ada pada *activity diagram*dapat dilihat pada tabel 2.4 di bawah ini :

Tabel 2. 4 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

Sumber : (Rosa & Shalahuddin, 2018)

2.6. Pengujian *Black Box*

Menurut Rosa dan Shalahuddin, (2018) mengatakan bahwa pengujian *Black-Box* yaitu menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian dimaksudkan untuk mengetahui apakah fungsi – fungsi, masukan dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan. Pengujian kotak hitam dilakukan dengan membuat kasus uji yang mencoba semua fungsi dengan memakai perangkat lunak, apakah sesuai dengan spesifikasi yang dibutuhkan. Kasus uji yang dibuat untuk melakukan pengujian *black-box* dengan kasus benar dan kasus salah. Adapun

kerangka yang akan digunakan untuk melakukan pengujian dapat dilihat pada Tabel 2.5.

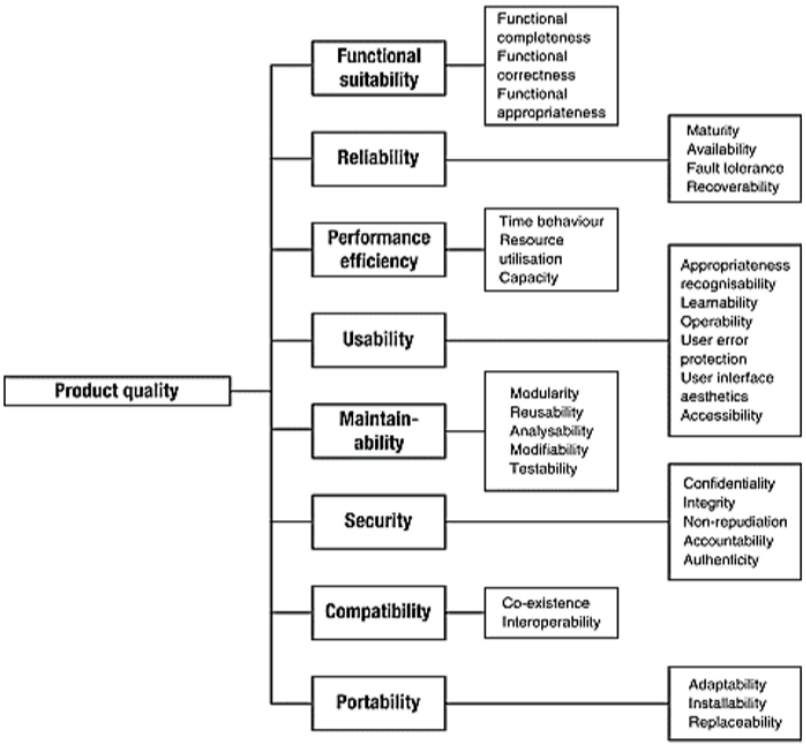
Tabel 2. 5 Kerangka Pengujian

Kasus dan Hasil Uji			
Data Masukan	**Data Diharapkan**	**Pengamatan**	**Kesimpulan**
			Diterima ( ) Ditolak ( )
			Diterima ( ) Ditolak ( )

2.7. Pengujian Sistem ISO 25010

ISO/IEC 25010 merupakan model kualitas sistem dan perangkat lunak yang menggantikan ISO/IEC 9126 tentang *software engineering* (Iqbal, 2016). 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
3. Data model kualitas



Gambar 2.3 Model kualitas produk ISO/IEC 25010

Adapun dimensi yang pertama terdapat beberapa faktor elemen diantaranya :

1. *Functionality* (Fungsionalitas). Kemampuan perangkat lunak Merupakan tingkatan dimana perangkat lunak dapat menyediakan fungsionalitas yang dibutuhkan ketika perangkat lunak digunakan pada kondisi spesifik tertentu dalam hal ini perangkat lunak dapat memenuhi kelayakan dari sebuah fungsi untuk melakukan pekerjaan yang spesifik bagi pengguna dan dapat memberikan hasil yang tepat dan ketelitian terhadap tingkat kebutuhan pengguna.
2. *Reliability* Merupakan tingkatan dimana perangkat lunak dapat bertahan pada tingkatan tertentu ketika digunakan oleh pengguna pada kondisi yang spesifik dalam hal ini perangkat lunak dapat beroperasi dan siap ketika dibutuhkan untuk digunakan dan juga dapat bertahan pada tingkat

kemampuan tertentu terhadap kegagalan, kesalahan serta perangkat lunak kembali pada tingkat tertentu dalam mengembalikan pengembalian data yang disebabkan kegagalan atau kesalahan pada perangkat lunak.

3. *Performance efficiency* Merupakan tingkatan dimana perangkat lunak dapat memberikan kinerja terhadap sejumlah sumber daya yang digunakan pada kondisi tertentu dalam hal ini *performance efficiency* dapat memberikan reaksi dan waktu yang dibutuhkan ketika melakukan aksi dari sebuah fungsi dan perangkat lunak dapat menggunakan sejumlah sumber daya ketika melakukan aksi dari sebuah fungsi.
4. *Usability* Perangkat lunak dapat dimengerti, dipelajari, digunakan dan menarik pengguna ketika digunakan dalam hal ini perangkat lunak mudah dipelajari oleh pengguna, perangkat lunak dapat digunakan dan dioperasikan oleh pengguna.
5. *Security* Merupakan perlindungan terhadap perangkat lunak dari berbagai ancaman atau keganjalan dalam hal ini perangkat lunak memiliki perlindungan terhadap data atau informasi dari pengguna dan merupakan dari kelengkapan, ketepatan dari sejumlah *asset* yang telah dijaga sehingga aksi atau tindakan yang dilakukan telah terbukti dan hal tersebut tidak dapat ditolak.
6. *Compability* Faktor ini merupakan kemampuan dari dua atau lebih komponen perangkat lunak dapat melakukan pertukaran informasi dan melakukan fungsi yang dibutuhkan ketika digunakan pada *hardware* atau lingkungan perangkat lunak yang sama.

7. *Maintainability* Merupakan tingkat dimana sebuah perangkat lunak dapat dimodifikasi. Dalam hal ini modifikasi adalah perbaikan, perubahan atau penyesuaian perangkat lunak untuk dapat berubah pada lingkungan, kebutuhan dan fungsionalitas yang spesifik. Selain itu perangkat lunak dapat dianalisi untuk mengetahui apa yang menyebabkan kegagalan pada perangkat lunak untuk mengidentifikasi bagian yang dapat dimodifikasi.
8. *Transferability* Merupakan kemudahan dimana sistem atau komponen dapat berpindah dari lingkungan satu ke lingkungan yang lain dalam hal ini perangkat lunak dapat beradaptasi dengan cepat pada spesifikasi lingkungan yang berbeda tanpa menerapkan aksi atau cara lain dari pada memberikan tujuan tertentu terhadap perangkat lunak yang telah ada.