

## BAB II LANDASAN TEORI

### 2.1 Tinjauan Studi

Tinjauan studi diperlukan sebagai bahan referensi bagi peneliti untuk menjadi pembanding antara penelitian yang dilakukan dengan penelitian terdahulu, serta untuk mempermudah dalam pengumpulan data dan metode analisis data yang digunakan dalam pengolahan data. Hasil penelitian terdahulu tidaklah semata-mata dijadikan sebagai acuan penulisan hasil dan pembahasan penelitian ini. Tinjauan studi memperlihatkan persamaan dan perbedaan beberapa hal, seperti metode, hasil, dan waktu penelitian. Penelitian terdahulu akan memberikan gambaran tentang penelitian sejenis yang akan dilakukan, sehingga dapat dijadikan referensi. Berikut ini merupakan tabel ulasan tentang penelitian terdahulu yang dijadikan sebagai referensi dalam penelitian ini.

**Tabel 2.1 Daftar Referensi**

1.	Penulis (Tahun)	Rohmat Indra Borman, Dyah Ayu Megawaty, dan Attohiroh (2020)
	Judul Penelitian	Implementasi Metode TOPSIS pada Sistem Pendukung Keputusan Pemilihan Biji Kopi Robusta yang Bernilai Mutu Ekspor (Studi Kasus: PT. Indo Cafco Fajar Bulan Lampung)
	Metode Penelitian	<i>Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS)</i>
	Hasil Penelitian	Studi ini menghasilkan aplikasi pendukung keputusan untuk memilih biji kopi dengan kualitas ekspor yang membantu pengambil keputusan memberikan rekomendasi hasil terbaik dan peringkat kelayakan kopi. Berdasarkan pengujian

		akurasi yang membandingkan hasil analisis pakar dengan sistem, didapatkan tingkat akurasi sebesar 84 persen dan tergolong kriteria baik.
2	Penulis (Tahun)	Adhie Thyo Priandika dan Agus Wantoro (2017)
	Judul Penelitian	Sistem Pendukung Keputusan Penerimaan Calon Siswa Baru pada SMK SMTI Bandar Lampung dengan Menggunakan Metode <i>Simple Additive Weighting</i> (SAW)
	Metode Penelitian	<i>Simple Additive Weighting</i> (SAW)
	Hasil Penelitian	Temuan penelitian ini adalah sistem pendukung keputusan penerimaan siswa baru dengan metode SAW yaitu pola perhitungan yang menggunakan penjumlahan berbobot penilaian kinerja setiap alternatif pada semua atribut dan dapat membantu pihak sekolah sebagai pengambil keputusan.
3	Penulis (Tahun)	Kelvin Julian Tannius, Jap Tji Beng, dan Dedi Trisnawarman (2019)
	Judul Penelitian	Sistem Pendukung Keputusan untuk Menentukan Biji Kopi Berkualitas Menggunakan <i>Simple Additive Weighting</i> (SAW)
	Metode Penelitian	<i>Simple Additive Weighting</i> (SAW)
	Hasil Penelitian	Hasil yang diperoleh adalah SPK yang dapat diakses oleh admin dan pelanggan untuk memilih biji kopi yang diinginkan. Sistem yang dibuat dapat membantu pelanggan dalam mengambil keputusan untuk menentukan kualitas biji kopi sesuai dengan kriteria yang mereka butuhkan dengan cepat.
4	Penulis (Tahun)	Ariawan Djoko Rachmato dan Jesica Andini Risanti (2019)
	Judul Penelitian	Sistem Pendukung Keputusan Kualitas Biji Kopi dengan Metode AHP ( <i>Analytical Hierarchy Process</i> ) Studi Kasus Cafe Kaki Bukit Lembang

	Metode Penelitian	<i>Analytical Hierarchy Process (AHP)</i>
	Hasil Penelitian	Hasil dari penelitian berupa suatu aplikasi yang dapat menentukan kualitas biji kopi dengan berdasarkan kriteria yang telah ditentukan, yaitu berupa nilai cacat, kadar air, warna, bau, dan ukuran biji kopi. Pemberian kriteria sangat membantu pengambil keputusan untuk menentukan biji kopi yang berkualitas.
5	Penulis (Tahun)	Muhammad Abidin, Rosmawati Tamin, dan UL Khairat (2020)
	Judul Penelitian	Penggunaan Metode <i>Simple Additive Weighting (SAW)</i> pada SPK Penentuan Biji Kopi Berkualitas (Studi Kasus: Petani Kopi Gisting - Lampung)
	Metode Penelitian	<i>Simple Additive Weighting (SAW)</i>
	Hasil Penelitian	Hasil perancangan dan implementasi SPK dalam penelitian ini adalah sistem dapat mengelola data kriteria penentuan biji kopi berkualitas serta data alternatif dan perbandingan. Selain itu, sistem dapat menghitung nilai pada masing-masing alternatif sehingga dapat menghasilkan keputusan biji kopi yang berkualitas.

## 2.2 Sistem Pendukung Keputusan

Konsep Sistem Pendukung Keputusan pertama kali diperkenalkan pada tahun 1970 oleh Michael S. Scott Morton dengan istilah “Sistem Keputusan Manajemen”. Kemunculan konsep ini membuat berbagai perusahaan dan universitas melakukan penelitian dan mengembangkan konsep sistem pendukung keputusan. Pada dasarnya, SPK dibuat untuk mendukung semua tahapan pengambilan keputusan, mulai dari identifikasi masalah, pemilihan data yang relevan, penentuan pendekatan

yang digunakan dalam proses pengambilan keputusan, hingga evaluasi opsi dari setiap alternatif (Setiyaningsih, 2015).

Menurut Scott dalam Setiyaningsih (2015), SPK adalah sistem komputasi interaktif yang mendukung pengambil keputusan dengan menggunakan data dan model keputusan untuk memecahkan masalah semi terstruktur dan tidak terstruktur, sehingga sangat meningkatkan efektivitas pengambil keputusan. Sementara itu, menurut Alavi dan Napier dalam Setiyaningsih (2015), SPK adalah seperangkat teknik pemrosesan data dan informasi yang dirancang menggunakan model yang sederhana, mudah dan adaptif untuk membantu dalam pengambilan keputusan.

### 2.3 Metode *Simple Additive Weighting* (SAW)

Metode SAW seringkali dikenal oleh beberapa orang dengan istilah metode penjumlahan terbobot. Konsep dasar metode SAW yaitu menjumlahkan bobot rating kinerja pada setiap alternatif dan atribut yang ada. Metode SAW membutuhkan proses normalisasi matriks keputusan (X) ke suatu skala, kemudian dibandingkan dengan rating seluruh alternatif yang diujikan (Setiyaningsih, 2015). Berikut ini adalah rumus dari metode Simple Additive Weighting (SAW):

$$r_{ij} = \begin{cases} \frac{X_{ij}}{\text{Max}^{X_{ij}}} \\ \frac{\text{Min}^{X_{ij}}}{X_{ij}} \end{cases} \dots\dots\dots (2.1)$$

Keterangan:

$r_{ij}$  = Rating kerja ternormalisasi

$\text{Max}^{ij}$  = Nilai maksimum dari setiap baris dan kolom

$\text{Min}^{ij}$  = Nilai minimum dari setiap baris dan kolom

$X_{ij}$  = Baris dan kolom dari matriks

Dimana  $r_{ij}$  adalah rating kinerja ternormalisasi dari alternatif  $A_i$  pada atribut  $C_j$ ;  $i = 1, 2, \dots, m$  dan  $j = 1, 2, \dots, n$ .

Nilai preferensi untuk setiap alternatif ( $V_i$ ) diberikan sebagai:

$$V_i = \sum_{j=1}^n w_j r_{ij} \dots\dots\dots (2.2)$$

Keterangan:

$V_i$  = Ranking untuk setiap alternatif

$W_j$  = Nilai bobot ranking (dari setiap kriteria)

$r_{ij}$  = Nilai rating kinerja ternormalisasi

Dimana nilai  $V_i$  yang lebih besar mengindikasikan bahwa alternatif  $A_i$  lebih terpilih.

Menurut Kusumadewi dalam Setiyaningsih (2015), terdapat beberapa penerapan metode Simple Additive Weighting (SAW) dalam pengambilan keputusan, yaitu:

1. Menentukan kriteria-kriteria yang dijadikan acuan dalam pendukung keputusan yaitu  $C_i$ .
2. Menentukan rating kecocokan setiap cara lain di setiap kriteria.
3. Membuat matriks keputusan berdasarkan kriteria ( $C_i$ ).
4. Melakukan normalisasi matriks berdasarkan persamaan yang disesuaikan dengan jenis atribut (*cost* dan *benefit*), lalu diperoleh matriks ternormalisasi  $R$ .
5. Hasil akhir diperoleh berupa perankingan yaitu penjumlahan dari perkalian matriks ternormalisasi  $R$  menggunakan vektor bobot sehingga diperoleh nilai terbesar yang dijadikan sebagai alternatif terbaik ( $A_i$ ).

## 2.4 Kriteria Pemilihan Biji Kopi Robusta

Data yang akan digunakan dalam menentukan biji kopi berkualitas berasal dari hasil observasi dan wawancara dengan empat kedai kopi di Bandar Lampung, yaitu Sakara Coffee, Rubik Coffee, Dr. Koffie, dan Bengkel Kopi. Berdasarkan hasil wawancara, terdapat persamaan dan perbedaan dalam menentukan biji kopi berkualitas.

### 1. Sakara Coffee

Biji kopi robusta Lampung yang biasa digunakan di Sakara Coffee berasal dari Kabupaten Lampung Barat dan Kabupaten Tanggamus. Kriteria yang digunakan Sakara Coffee dalam menentukan biji kopi robusta berkualitas adalah kadar air, ukuran, aroma dan nilai cacat. Dalam menentukan kualitas *green bean*, Sakara Coffee masih menggunakan cara manual dengan mengandalkan indra penglihatan, penciuman dan peraba. Selain itu, Sakara Coffee masih menggunakan informasi dari petani. Dalam menentukan biji kopi berkualitas, masalah yang dihadapi Sakara Coffee adalah cuaca pada saat pascapanen yang berubah dan proses pengelolaan pascapanen biji kopi yang dilakukan oleh petani dari tahun ke tahun berbeda.

### 2. Rubik Coffee

Biji kopi robusta Lampung yang biasa digunakan di Rubik Coffee berasal dari Kabupaten Lampung Barat, Kabupaten Tanggamus, dan Kabupaten Pesawaran. Kriteria yang digunakan Rubik Coffee dalam menentukan biji kopi robusta berkualitas adalah kadar air, kotoran, kopi segar, ukuran, aroma dan nilai cacat. Dalam menentukan kualitas *green bean*, Rubik Coffee masih menggunakan cara manual dengan mengandalkan indra penglihatan,

penciuman dan peraba. Selain itu, Rubik Coffee masih menggunakan informasi dari petani. Dalam menentukan biji kopi berkualitas, masalah yang dihadapi Rubik Coffee adalah cuaca pada saat pasca panen yang berubah dan proses pengelolaan pasca panen biji kopi yang dilakukan oleh petani dari tahun ke tahun berbeda.

### 3. Dr. Koffie

Biji kopi robusta Lampung yang biasa digunakan di Dr. Koffie berasal dari Kabupaten Lampung Barat, dan Kabupaten Tanggamus. Kriteria yang digunakan Dr. Koffie dalam menentukan biji kopi robusta berkualitas adalah nilai cacat, ukuran dan kadar air. Dalam menentukan biji kopi berkualitas Dr. Koffie melakukan pembinaan kepada petani yang menyuplai *green bean* ke Dr. Koffie. Dalam menentukan biji kopi berkualitas, masalah yang dihadapi Dr. Koffie adalah cuaca pada saat pasca panen yang berubah.

### 4. Bengkel Kopi

Biji kopi robusta Lampung yang biasa digunakan di Bengkel Kopi berasal dari Kabupaten Lampung Barat, Kabupaten Tanggamus dan Kabupaten Pesawaran. Kriteria yang digunakan Bengkel Kopi dalam menentukan biji kopi robusta berkualitas adalah kadar air, serangga hidup, ukuran dan nilai cacat. Dalam menentukan kualitas *green bean* Bengkel Kopi masih menggunakan cara manual dengan mengandalkan indra penglihatan, penciuman dan peraba. Selain itu, Bengkel Kopi masih menggunakan informasi dari petani. Dalam menentukan biji kopi berkualitas, masalah yang dihadapi Bengkel Kopi adalah cuaca pada saat pasca panen yang berubah dan proses pengelolaan pasca panen biji kopi yang dilakukan oleh petani dari tahun ke tahun berbeda.

Berdasarkan hasil wawancara dengan empat kedai kopi tersebut, maka biji kopi robusta Lampung yang akan dijadikan sebagai alternatif dalam penelitian ini berasal dari Kabupaten Lampung Barat (Kecamatan Sekincau), Kabupaten Tanggamus (Kecamatan Ulu Belu), dan Kabupaten Pesawaran (Kecamatan Gedong Tataan). Kriteria yang digunakan yaitu kadar air (C1) dengan syarat kadar air maksimal 13%, aroma (C2) dengan syarat tidak berbau busuk, ukuran (C3) dengan syarat biji kopi berukuran serupa, serta nilai cacat (C4) dengan syarat nilai cacat maksimal 0,5%. Berikut adalah tabel kriteria pemilihan biji kopi robusta beserta nilai mutunya.

**Tabel 2.2 Kriteria Pemilihan Biji Kopi Robusta**

No.	Kode Kriteria	Kriteria	Nilai Mutu	Nilai
1	C1	Kadar Air	0-13%	1
			14%-15%	2
			>15%	3
2	C2	Aroma	Berbau Busuk	1
			Tidak Berbau Busuk	2
3	C3	Ukuran	Tidak Serupa	1
			Serupa	2
4	C4	Nilai Cacat	0-0,5%	1
			0,6%-0,7%	2
			>0,7%	3

## 2.5 Website

*Website* dapat diartikan sebagai kumpulan halaman yang berisi informasi data digital baik berupa teks, gambar, animasi, suara dan audio atau gabungan dari semuanya yang disediakan melalui jalur koneksi internet sehingga dapat diakses dan dilihat oleh semua orang di seluruh dunia. Halaman *website* dibuat menggunakan bahasa standar yaitu HTML. Skrip HTML ini akan diterjemahkan oleh *web browser* sehingga dapat ditampilkan dalam bentuk informasi yang dapat



dibaca oleh semua orang (Abdulloh, 2018:1). Selanjutnya menurut Abdullah (2018:1-2), secara umum, *website* dibagi menjadi 3 jenis, yaitu *website* statis, *website* dinamis, dan *website* interaktif.

### **2.5.1 Website Statis**

*Website* statis yaitu jenis *website* yang isinya tidak diperbaharui secara berkala, sehingga isinya dari waktu ke waktu akan selalu tetap. *Website* jenis ini biasanya hanya digunakan untuk menampilkan profil dari pemilik *website* seperti profil perusahaan atau organisasi.

### **2.5.2 Website Dinamis**

*Website* dinamis yaitu jenis *website* yang isinya terus diperbaharui secara berkala oleh pengelola web atau pemilik *website*. *Website* jenis ini banyak dimiliki oleh perusahaan atau perorangan yang aktivitas bisnisnya berkaitan dengan internet. Contoh dari *website* jenis ini yaitu *web blog* dan *website* berita.

### **2.5.3 Website Interaktif**

*Website* interaktif pada dasarnya termasuk dalam kategori *website* dinamis, dimana isi informasinya selalu diperbaharui dari waktu ke waktu. Hanya saja, isi informasi tidak hanya diubah oleh pengelola *website* tetapi lebih banyak dilakukan oleh pengguna *website* itu sendiri. Contoh *website* jenis ini yaitu *website* jejaring sosial seperti Facebook dan Twitter atau *website marketplace* seperti Bukalapak, Tokopedia, dan sebagainya.

## 2.6 Database

Menurut Abdulloh (2018:103), *database* atau basis data, adalah kumpulan informasi yang disimpan dalam komputer secara sistematis sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi. Dalam pembuatan aplikasi perlu memperhatikan rancangan *database* agar aplikasi yang dibuat dapat berjalan sesuai konsep yang direncanakan.

Menurut Martin dalam Sutabri (2016:135), *database* adalah suatu perpaduan data terhubung (*interrelated data*) yang disimpan secara bersama-sama pada suatu media, tanpa mengatap satu sama lain atau tidak perlu suatu kerangkapan data (*controlled redundancy*) dengan cara tertentu sehingga mudah dipergunakan atau ditampilkan balik, bisa digunakan oleh satu atau lebih program *software* secara optimal, data disimpan tanpa mengalami ketergantungan di program yang akan menggunakannya, data disimpan sedemikian rupa sehingga penambahan, pengambilan, dan modifikasi dapat dilakukan dengan praktis dan terkendali.

## 2.7 MySQL

Menurut Pamungkas (2017:79), MySQL adalah sebuah perangkat lunak *open source* untuk sistem manajemen basis data SQL (*database management system*) atau DBMS yang *multithread multiuser*. Berikut merupakan keunggulan yang dimiliki oleh basis data MySQL:

### 1. Portabilitas

MySQL dapat berjalan stabil pada berbagai sistem operasi seperti Windows, Linux, FreeBSD, Mac OS X Server, Solaris, Amiga, dan lain sebagainya.

## 2. *Open Source*

MySQL didistribusikan sebagai perangkat lunak sumber terbuka, di bawah lisensi GPL sehingga dapat digunakan secara gratis.

## 3. *Multiuser*

MySQL dapat digunakan oleh beberapa pengguna dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.

## 4. *Performance Tuning*

MySQL memiliki kecepatan yang menakjubkan dalam menangani *query* sederhana, dengan kata lain dapat memproses lebih banyak SQL persatuan waktu.

## 5. Ragam Tipe Data

MySQL memiliki ragam tipe data yang sangat kaya, seperti *signed/unsigned integer, float, double, char, text, date, timestamp*, dan lain-lain.

## 6. Perintah dan Fungsi

MySQL memiliki operator dan fungsi secara penuh yang mendukung perintah *Select* dan *Where* dalam perintah (*query*).

## 7. *Security* (Keamanan)

MySQL memiliki beberapa lapisan keamanan seperti *level subnet mask*, nama *host*, dan izin akses pengguna dengan sistem perizinan yang mendetail serta sandi terenkripsi.

## 8. Skalabilitas dan Pembatasan

MySQL mampu menangani basis data dalam skala esar, dengan jumlah rekaman lebih dari 50 juta dan 60 ribu tabel serta 5 milyar baris. Selain itu batas indeks yang dapat ditampung mencapai 32 indeks pada tiap tabelnya.

## 9. Konektivitas

MySQL dapat melakukan koneksi dengan klien menggunakan protokol TCP/IP, Unix Sockets (UNIX), atau Named Pipes (NT).

## 10. Lokalisasi

MySQL dapat mendeteksi pesan kesalahan pada klien dengan menggunakan lebih dari dua puluh bahasa. Meskipun demikian, Bahasa Indonesia belum termasuk di dalamnya.

### **2.8 *Hypertext Markup Language (HTML)***

HTML merupakan singkatan *Hypertext Markup Language* yaitu bahasa standar web yang dikelola penggunaannya W3C (*World Wide Web Consortium*) berupa tag-tag yang menyusun setiap elemen dari website. HTML berperan sebagai penyusun struktur halaman *website* yang menempatkan setiap elemen *website* sesuai *layout* yang diinginkan (Abdulloh, 2018:7).

Selanjutnya menurut Abdulloh (2018:7), HTML biasanya disimpan dalam sebuah *file* berekstensi *.html*. Untuk mengetikkan skrip HTML dapat menggunakan *text editor* seperti Notepad sebagai bentuk paling sederhana atau *text editor* khusus yang dapat mengenali setiap unsur skrip HTML dan menampilkannya dengan warna yang berbeda sehingga mudah dibaca, seperti Notepad++, Sublime Text, dan masih banyak lagi aplikasi yang sejenis.

### **2.9 *Hypertext Processor (PHP)***

Menurut Pamungkas (2017:38), PHP (*Hypertext Preprocessor*) yaitu bahasa pemrograman yang digunakan secara luas untuk penanganan pembuatan dan pengembangan sebuah situs web dan bisa digunakan bersamaan dengan HTML.

PHP bersifat *server-side*, artinya berbentuk *script* yang disimpan dan dijalankan di komputer *server* (*WebServer*) sedang hasilnya yang dikirimkan ke komputer *client* (*WebBrowser*) dalam bentuk *script* HTML (*Hypertext Mark Up Language*).

PHP merupakan kependekan dari *PHP Hypertext Preprocessor* atau bahasa pemrograman web yang dapat disisipkan dalam skrip HTML dan bekerja di sisi server. Tujuan dari bahasa ini adalah membantu para pengembangan web untuk membuat web dinamis dengan cepat (Abdulloh, 2018:127). Sedangkan menurut Wiswakarma (2009:12), PHP atau *Hypertext Preprocessor* ialah salah satu jenis bahasa pemrograman web yang bersifat *open source*, sebagai akibatnya dapat dipergunakan oleh siapapun secara gratis.

## **2.10 Framework CodeIgneter**

CodeIgniter ialah kerangka pengembangan *software*, yang bisa digunakan buat membuat *website* menggunakan PHP, yang bersifat *open source framework* dengan mempunyai serangkaian fungsi yang banyak untuk meningkatkan kecepatan kerja pada pembuatan *website*. CodeIgniter mempunyai kelebihan dibanding *framework* web lainnya yaitu sangat terstruktur, ringa dan praktis untuk dipelajari, dokumentasi yang lengkap, serta didukung oleh banyaknya forum CodeIgniter (Nawawi, dkk, 2020:3). CodeIgniter merupakan *framework* PHP yang memakai konsep MVC dalam penerapannya.

*Model-View-Controller* (MVC) merupakan suatu metode yang memisahkan *data logic* (*model*) yang berasal dari *presentation logic* (*view*) serta *process logic* (*controller*) atau secara sederhana ialah memisahkan antara desain antarmuka, data, serta proses (Endra dan Aprilita, 2018).

### 1. *Model*

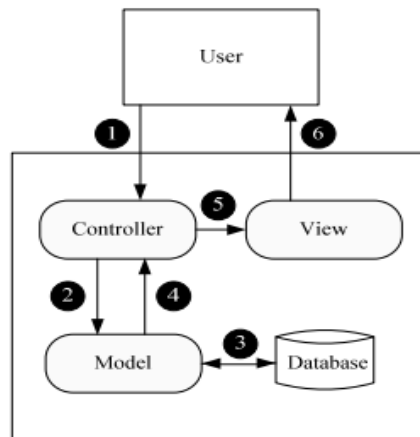
Model mengelola basis data (RDBMS) mirip MySQL ataupun Oracle RDMS. Model terkoneksi dengan *databases* dan biasanya pada model akan berisi *class* ataupun fungsi untuk membuat (*create*), memperbarui data (*update*), menghapus data (*delete*), mencari data (*search*), serta memilih data (*select*), pada *database*. Kemudian, model akan terkoneksi dengan perintah-perintah (*query*) menjadi respon fungsi-fungsi (*create, update, delete* dan *select*).

### 2. *View*

*View* adalah bagian dari antarmuka pengguna atau bagian yang nantinya adalah tampilan untuk *end-user*. *View* bisa berupa laman HTML, CSS, RSS, Javascript JQuery, Ajax, dan sebagainya. Metode yang dipakai merupakan metode MVC sehingga dalam *view* tidak boleh ada pemrosesan data ataupun pengaksesan yang berhubungan dengan *database*. *View* hanya menampilkan data-data hasil dari perintah *model* dan *controller*.

### 3. *Controller*

*Controller* ialah penghubung antara *view* dan *model*, maksudnya adalah sebab *model* tidak dapat terkoneksi langsung dengan *view* ataupun sebaliknya. Jadi, *controller* inilah yang berperan menjadi jembatan antar keduanya. Tugas *controller* ialah sebagai pemrosesan data, menyediakan variabel yang akan ditampilkan oleh *view*, lalu memanggil *model* sehingga mampu mengakses *databases, error handling, validasi* atau *check* terhadap suatu masukan (*input*).

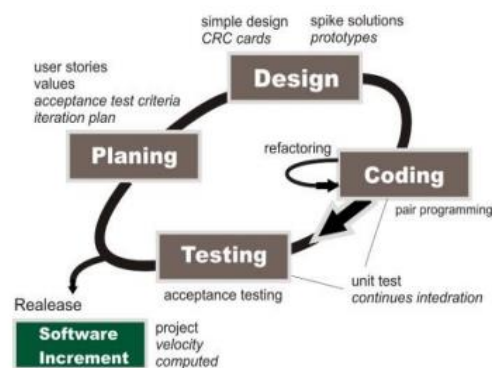


**Gambar 2.1 Alur MVC**

Sumber: Supaartagorn dalam Suendri (2018)

### 2.11 Metode Pengembangan Sistem *Extreme Programming* (XP)

*Extreme Programming* (XP) adalah metode pengembangan sistem yang digunakan dalam penelitian ini. Berdasarkan pernyataan Septilia dan Styawati (2020), *Extreme Programming* (XP) merupakan metodologi pengembangan *software* yang ditujukan untuk menaikkan kualitas *software* dan tanggap terhadap perubahan kebutuhan pelanggan. Jenis pengembangan *software* seperti ini dimaksudkan untuk mempertinggi produktivitas serta memperkenalkan pos pemeriksaan dimana persyaratan pengguna baru bisa diadopsi. *Extreme Programming* terdiri dari empat tahapan yang dapat dilihat pada Gambar 2.2.



**Gambar 2.2 *Extreme Programming***

Sumber: Septilia dan Styawati, 2020

### 1. *Planning* (Perencanaan)

Pada tahapan ini, peneliti melakukan perencanaan dalam pembuatan Sistem Pendukung Keputusan (SPK) biji kopi berkualitas berbasis *website* dengan mengumpulkan data dan menganalisis kebutuhan-kebutuhan yang diperlukan oleh Sakara Coffee untuk kemudian dikembangkan menjadi sebuah sistem. Aktivitas yang dilakukan pada saat pengumpulan data adalah mengidentifikasi permasalahan dalam penentuan biji kopi berkualitas dengan cara wawancara terhadap pemilik dan barista Sakara Coffee dan beberapa kedai kopi yang ada, serta melakukan observasi. Selain itu peneliti juga melakukan studi literatur, dengan tujuan untuk memperkuat dasar-dasar teoritis sebagai dasar penyelesaian masalah yang diperoleh dari berbagai referensi buku dan jurnal ilmiah yang relevan dengan judul penelitian. Analisis kebutuhan sistem merupakan tahapan awal dan utama untuk membuat pondasi dalam pengembangan sistem. Kebutuhan sistem yang ditentukan oleh peneliti terdiri dari kebutuhan fungsional dan non fungsional.

### 2. *Design* (Perancangan)

Tahap berikutnya adalah membuat perancangan dimana pada tahapan ini dilakukan kegiatan pemodelan sistem dengan menggunakan diagram *Unified Modelling Language* (UML) yang terdiri dari *use case diagram*, *class diagram*, dan *activity diagram* menggunakan perangkat lunak Astah Professional. Selain itu, peneliti juga membuat rancangan antarmuka (*interface*) menggunakan perangkat lunak Balsamiq Mockups 3, yang merupakan rancangan tampilan visual *website* yang menjembatani sistem dengan *user* (*owner* dan barista Sakara Coffee).



### 3. *Coding* (Pengkodean)

Tahapan ini merupakan kegiatan implementasi pemodelan yang sudah dibuat pada tahap perancangan ke dalam bentuk *user interface* dengan menggunakan bahasa pemrograman *Hypertext Preprocessor* (PHP). Pada tahapan *coding*, peneliti menggunakan Sublime Text 3 untuk menulis *source code* yang nantinya akan menjadi Sistem Pendukung Keputusan berbasis *website*. Selain itu, peneliti juga membuat basis data (*database*) menggunakan perangkat lunak MySQL. Sistem dijalankan dengan menggunakan XAMPP sebagai *localhost*.

### 4. *Testing* (Pengujian)

Setelah tahapan *coding* selesai, kemudian dilakukan tahapan pengujian sistem untuk mengetahui kesalahan apa saja yang muncul saat *website* dioperasikan oleh *owner* dan barista Sakara Coffee, serta apakah komponen-komponen pada sistem yang dikembangkan telah sesuai dengan fungsinya. Metode pengujian yang digunakan pada penelitian ini adalah *black box testing*. Pengujian ini dilakukan terhadap *form* beberapa *output* apakah sudah berjalan sesuai dengan fungsinya masing-masing. Jika ditemukan kesalahan, maka sistem dapat diperbaiki.

## 2.12 *Unified Modeling Language* (UML)

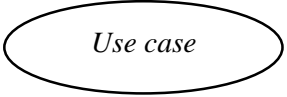
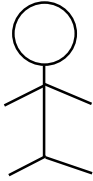

*Unified Modeling Language* (UML) adalah bahasa standar yang banyak digunakan di dunia industri untuk mendefinisikan persyaratan, membuat analisis dan desain, dan menggambarkan arsitektur dengan cara berorientasi objek. UML adalah bahasa visual untuk memodelkan dan mengkomunikasikan informasi sistem melalui diagram dan teks pendukung. UML muncul dari kebutuhan pemodelan visual untuk menentukan, mendeskripsikan, membangun, dan mendokumentasikan

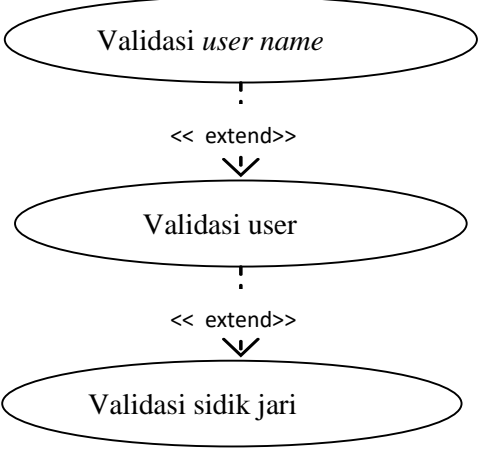
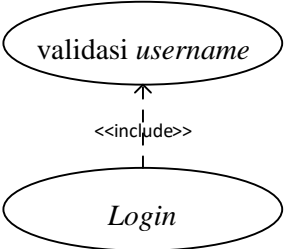
sistem perangkat lunak. UML hanya digunakan untuk pemodelan. Akibatnya, meskipun UML paling umum digunakan dalam metodologi objek, penerapannya tidak terbatas pada metodologi itu (Rosa A.S dan M. Shalahudin, 2014:133).

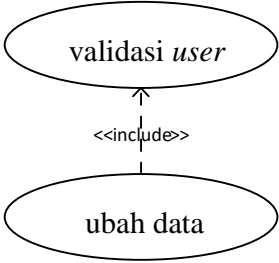
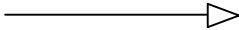
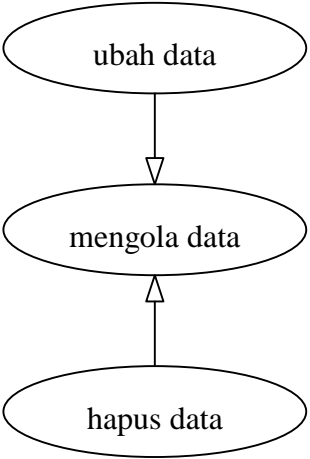
### 2.12.1 Use Case Diagram

Diagram *use case* merupakan pemodelan untuk aktivitas sistem informasi yang akan dibuat. Syarat penamaan pada *use case* adalah nama yang didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case*. Simbol-simbol pada *use case diagram* dapat dilihat pada Tabel 2.3.

**Tabel 2.3 Use Case Diagram**

Nama Simbol	Simbol	Deskripsi
<i>Use case</i>		Fungsionalitas yang disediakan sistem menjadi unit-unit yang saling bertukar pesan, umumnya dinyatakan memakai kata kerja di awal frase nama <i>use case</i> .
Aktor/ <i>Actor</i>		Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dirancang. Walaupun simbol berasal dari aktor yang bergambar orang, akan tetapi aktor belum tentu mengidentifikasi user. Umumnya dinyatakan memakai kata benda pada awal frase nama aktor.
<i>Asosiasi/Association</i>		Komunikasi antara aktor dan <i>use case</i> yang berpartisipasi pada sistem atau memiliki interaksi dengan aktor.

Ekstensi/ <i>Extend</i>	<p>-- &lt;&lt;extend&gt;&gt; --&gt;</p>	<p>Penghubung <i>use case</i> tambahan dimana <i>use case</i> yang ditambahkan bisa berdiri sendiri, seperti menggunakan prinsip inheritance pada pemrograman berorientasi objek, umumnya <i>use case</i> tambahan mempunyai nama depan yang sama dengan <i>use case</i> yang ditambahkan, misal arah panah menunjuk pada <i>use case</i> yang ditambahkan, umumnya <i>use case</i> yang menjadi ekstensinya ialah jenis yang sama dengan <i>use case</i> yang menjadi induknya.</p>  <pre> graph TD     A([Validasi user name])     B([Validasi user])     C([Validasi sidik jari])     B -.-&gt; &lt;&lt;extend&gt;&gt;  A     C -.-&gt; &lt;&lt;extend&gt;&gt;  B   </pre>
Menggunakan/ <i>Include/Uses</i>	<p>&lt;&lt;include&gt;&gt; →</p>	<p><i>Include</i> berarti <i>use case</i> yang ditambahkan akan selalu merespon saat <i>use case</i> tambahan dijalankan, misalnya:</p>  <pre> graph TD     A([validasi username])     B([Login])     A --&gt; &lt;&lt;include&gt;&gt;  B   </pre>


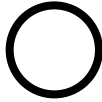
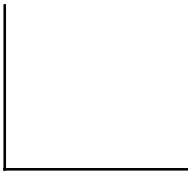
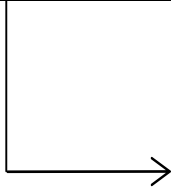
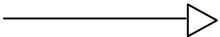
		<p>- <i>Include</i> berarti <i>use case</i> yang ditambahkan akan selalu melakukan pengecekan apakah <i>use case</i> yang ditambahkan telah dijalankan, misalnya:</p>  <pre> graph BT     A(ubah data) -- &lt;&lt;include&gt;&gt; --&gt; B(validasi user)   </pre> <p>Kedua definisi di atas dapat diterapkan pada salah satu atau keduanya, tergantung pada interpretasi yang dibutuhkan.</p>
<p>Generalisasi/<i>Generalization</i></p>		<p>Hubungan generalisasi-spesialisasi (umum-khusus) antara dua <i>use case</i> dimana salah satunya memiliki fungsi yang lebih umum daripada yang lainnya, misalnya:</p>  <pre> graph TD     A(ubah data) --&gt; B(mengola data)     C(hapus data) --&gt; B   </pre>

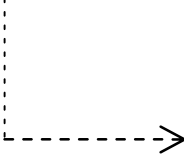
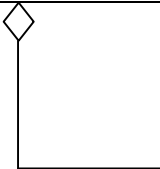
		Arah panah mengarah pada <i>use case</i> yang menjadi generalisasinya (umum).
--	--	---

### 2.12.2 Class Diagram

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk merancang sistem. Kelas mempunyai atribut pola dan metode atau operasi. Simbol-simbol pada *class diagram* dapat dilihat pada Tabel 2.4.

**Tabel 2.4 Class Diagram**


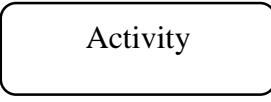
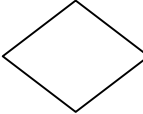
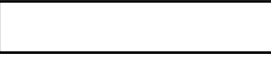
Nama Simbol	Simbol	Deskripsi
Kelas/ <i>Class</i>		Kelas pada struktur sistem.
Antarmuka/ <i>Interface</i>		Sama dengan konsep antarmuka dalam pemrograman yang berorientasi objek.
Asosiasi/ <i>Association</i>		Relasi antar kelas dengan makna umum, asosiasi biasanya beragam.
Isosiasi Berarah/ <i>Directed association</i>		Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, isosiasi biasanya juga beragam.
Generalisasi/ <i>Generalization</i>		Relasi antar kelas dengan makna generalisasi-spesialisasi.


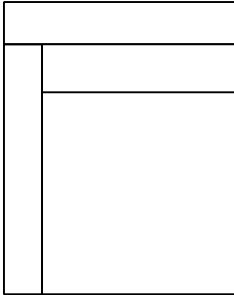
<i>Dependence</i>		Relasi antar kelas dengan makna kebergantungan antar kelas.
<i>Agresi/Aggregation</i>		Relasi antar kelas dengan makna seluruh bagian ( <i>whole-part</i> ).

### 2.12.3 Activity Diagram

*Activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem yang terdapat pada *software*. Hal yang perlu diperhatikan disini adalah bahwa *activity diagram* bukan menggambarkan apa yang dilakukan aktor melainkan aktivitasnya. Simbol-simbol pada *Activity Diagram* dapat dilihat pada Tabel 2.5.

**Tabel 2.5 Activity Diagram**

<b>Nama Simbol</b>	<b>Simbol</b>	<b>Deskripsi</b>
Status Awal/ <i>Initial node</i>		Sebuah diagram aktivitas yang memiliki status awal.
Aktivitas/ <i>Activity</i>		Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
Percabangan/ <i>Decision</i>		Asosiasi percabangan dimana jika pilihan aktivitas lebih dari satu.
Penggabungan/ <i>Join</i>		Asosiasi penggabungan dimana lebih dari satu

		aktivitas digabungkan menjadi satu.
Status Akhir/ <i>Activity Final</i>		Status akhir yang dilakukan oleh sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
<i>Swimlane</i>		Memisahkan aktor yang bertanggung jawab terhadap aktivitas yang terjadi.

### 2.13 XAMPP

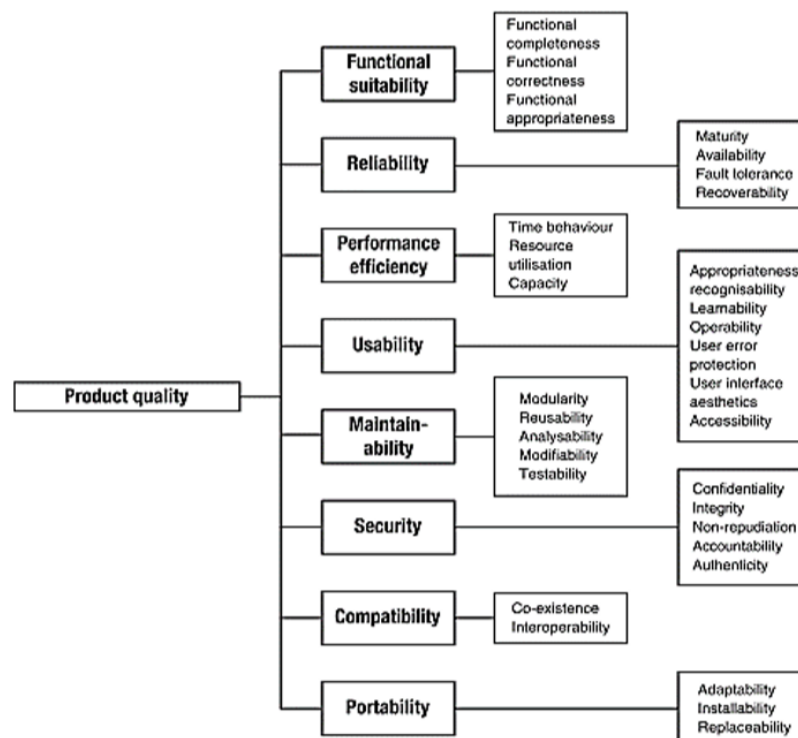
XAMPP merupakan *software* yang mendukung berbagai sistem operasi atau kompilasi yang berasal dari beberapa program. XAMPP berfungsi menjadi server yang mandiri (*localhost*), yang terdiri atas program Apache, HTTP, MySQL *database*, dan penerjemah bahasa yang ditulis dalam bahasa pemrograman PHP dan Perl. Program ini tersedia pada GNU (*General Public License*) dan dapat diperoleh secara gratis (*free*). XAMPP merupakan *web server* yang praktis untuk digunakan dan dapat melayani tampilan laman web yang dinamis (Nurcholish, 2018:23).

Haviluddin, dkk (2016:1) mengatakan bahwa XAMPP merupakan *software* yang merangkap beberapa *software* lain yang dibutuhkan pada pengembangan web. Nama XAMPP merupakan singkatan yang berasal dari *software* utama yang ada di dalamnya; X (alfabet X berarti *cross-platform*, dimana perangkat lunak XAMPP tersedia untuk banyak Sistem Operasi), A (Apache web server), M (MySQL), P

(PHP), dan P (Perl). Selain *software* tersebut, XAMPP juga menyertakan modul lain seperti OpenSSL dan PhpMyAdmin.

#### 2.14 Pengujian Sistem ISO/IEC 25010

Proses pengujian perangkat lunak sangat diperlukan dalam pengembangan suatu sistem. Pengujian bertujuan untuk memahami letak kesalahan atau *error* yang terjadi dan muncul pada sistem yang sedang dikembangkan. Pengujian ISO/IEC 25010 merupakan model kualitas sistem dan perangkat lunak yang menggantikan ISO/IEC 9126 tentang *software engineering* (Iqbal, 2016). ISO/IEC 25010 dibuat oleh ISO (*International Organization for Standardization*) dan IEC (*International Electrotechnical Commission*). *Product quality* ini juga digunakan untuk tiga model kualitas yang berbeda untuk produk perangkat lunak antara lain kualitas dalam model penggunaan, model kualitas produk, dan data model kualitas.



**Gambar 2.3 Model kualitas produk ISO/IEC 25010**

Sumber: Iqbal (2016)



Seperti yang diketahui pada gambar di atas, pengujian ISO/IEC 25010 memiliki delapan karakteristik, yaitu:

1. *Functionality*

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*

Karakteristik ini menguji 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*

Karakteristik ini menguji 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*

Karakteristik ini 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*

Karakteristik ini 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 dianalisis untuk mengetahui apa yang menyebabkan kegagalan pada perangkat lunak untuk mengidentifikasi bagian yang dapat dimodifikasi.

#### 8. *Portability*

Karakteristik ini menguji 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.