

## BAB II LANDASAN TEORI

### 2.1. Tinjauan Pustaka

Penelitian membutuhkan studi literatur yang menjadi landasan teori sehingga dapat mendukung untuk melakukan perancangan sistem. Berdasarkan studi literatur yang dilakukan, terdapat beberapa penelitian terkait sebagai dasar penelitian ini sebagai berikut.

Penelitian Yoraeni, dkk. (2020) perancangan sistem pendaftaran siswa baru berbasis *web* pada sekolah Mi Daarul Hikmah agar menghasilkan informasi yang cepat, akurat, dan tepat. Pada tahapan pembuatan desain menggunakan struktur navigasi, *LRS (Logical Record Structure)* untuk menentukan kardinalitas, jumlah *table*, dan *foreign key*, *ERD (Entity Relationship Diagram)* untuk mengimplementasikan basis data secara logika dan secara fisik. Serta *XAMPP* sebagai *web server*, *database* untuk menyimpan data-data yang digunakan pada aplikasi ini. Perancangan sistem dengan menggunakan Bahasa pemrograman *PHP*, *Javascript*, *HTML*, *CSS*, dan untuk mengatur tampilan *editor* menggunakan *Sublime Text*. Keseluruhan data dapat tersimpan kedalam *database* yang akan menjadi arsip pihak sekolah akan jauh lebih aman dibandingkan harus melakukan penyimpanan berkas laporan yang tidak efisien dan memungkinkan kerusakan dan kehilangan. Sistem informasi ini dapat menampilkan pengumuman calon siswa baru yang diterima maupun ditolak.

Penelitian Anisah dan Sayuti (2018) perancangan sistem pendaftaran *online* untuk penerimaan siswa baru SMK Negeri 1 Kelapa Bangka Barat berbasis *website* menggunakan metode *Waterfall*. Untuk analisis dan perancangan secara bertahap menggunakan metode berorientasi objek dengan menggunakan *diagram UML*. Pada tahap analisis menggunakan diagram aktivitas dan diagram *use case*, pada tahap perancangan menggunakan *class diagram*. Perancangan sistem ini dapat mempermudah proses penerimaan siswa baru, penyimpanan data terkait penerimaan siswa baru dapat disimpan dengan baik melalui *database*, dan informasi mengenai penerimaan siswa baru dapat diperoleh secara akurat dengan

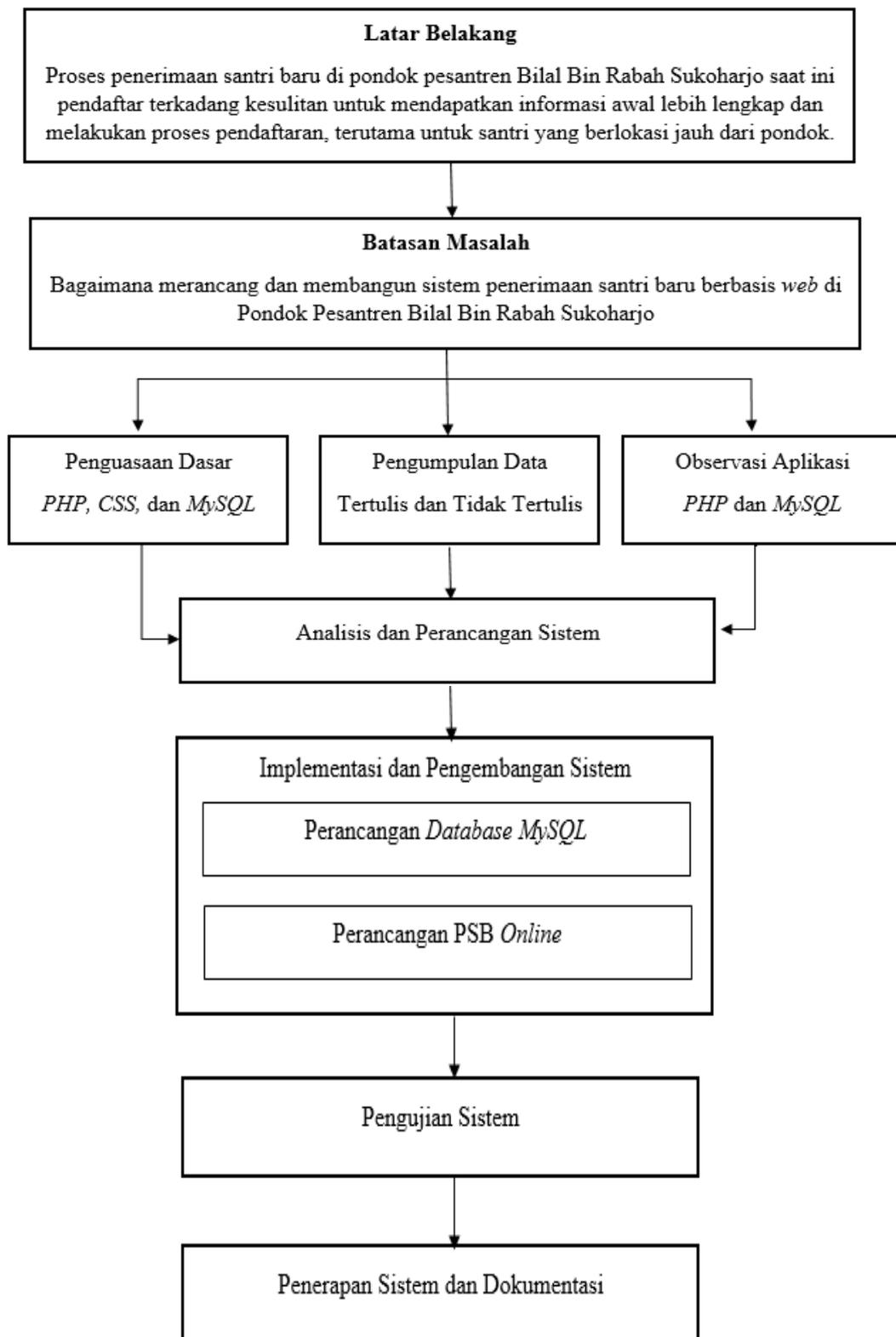
mempertimbangkan pengambilan keputusan.

Penelitian Najamudin, dkk. (2019) dibangun aplikasi pendaftaran peserta didik baru berbasis *web* pada SMK Negeri 2 Kuripan. Alur dan data mengacu pada sistem konvensional yang sedang diterapkan. Metode pengumpulan data dilakukan adalah observasi, wawancara, analisis, perancangan sistem dan pengujian. Perancangan menggunakan *UML* untuk menggambarkan alur penggunaan aplikasi, *interface*, dan struktur menu. Implementasi aplikasi menerapkan *CodeIgniter* sedangkan untuk menjalankan aplikasi menggunakan lokal *hosting XAMPP*. Hasil pengujian menggunakan *Black-box* menunjukkan bahwa aplikasi dapat berjalan sesuai kebutuhan yaitu dapat melakukan pendaftaran dengan *form* biodata dan pilihan jurusan, data peserta didik tersimpan pada *database*, pengelolaan seleksi serta menampilkan pengumuman hasil seleksi. Aplikasi dapat menghasilkan dokumen cetak beberapa *form* untuk kebutuhan arsip dokumen.

Penelitian Muslihudin dan Imamudin (2019) perancangan dan pengembangan program aplikasi sistem informasi pendaftaran siswa baru berbasis *web mobile* pada SMA Negeri 1 Ulu Belu. Metode yang digunakan untuk pembuatan sistem informasi menggunakan permodelan *Entity Relationship Diagram (ERD)*. Hasil akhir dari perancangan dan pengembangan sistem informasi tersebut adalah adanya jaringan seluler berbasis *web* yang dapat memberikan kemudahan akses informasi dan proses pendaftaran, sehingga membuat proses penerimaan siswa baru menjadi lebih efektif dan efisien.

## **2.2.Kerangka Pemikiran**

Berikut ini adalah tahapan kerangka pemikiran dalam melakukan perancangan dan pembuatan sistem penerimaan santri baru berbasis *web* pondok pesantren Bilal Bin Rabah. Kerangka pemikiran dapat dilihat pada Gambar 2.1.



Gambar 2.1. Kerangka Pemikiran

Penjelasan dari kerangka pemikiran pada Gambar 2.1 adalah sebagai berikut:

1) Latar Belakang Masalah

Proses penerimaan santri baru di Pondok Pesantren Bilal Bin Rabah Sukoharjo saat ini pendaftar terkadang kesulitan untuk mendapatkan informasi awal lebih lengkap dan melakukan proses pendaftaran, terutama untuk santri yang berlokasi jauh dari pondok pesantren. Jika jumlah pendaftar semakin banyak, akan membuat pihak pondok pesantren kesulitan dalam mengelola pendaftar yang mendaftar melalui *Whatsapp*. Selain itu proses administrasi penerimaan santri baru cenderung lambat, karena data santri baru yang telah mendaftar belum terintegrasi dan terkelola dengan baik.

2) Rumusan Masalah

Bagaimana merancang dan membangun sistem penerimaan santri baru berbasis *web* menggunakan *CodeIgniter 4* pada Pondok Pesantren Bilal Bin Rabah Kabupaten Sukoharjo.

3) Penguasaan Dasar

Penguasaan dasar *PHP*, dan *MySQL* merupakan suatu kunci terciptanya sistem penerimaan santri baru pada Pondok Pesantren Bilal Bin Rabah Sukoharjo berbasis *web*, karena pembuatan sistem ini menggunakan bahasa pemrograman *PHP* dan menggunakan *database MySQL* sebagai media penyimpanan data.

4) Pengumpulan Data

Tahap pengumpulan data pada penelitian ini melalui observasi, wawancara dan studi literatur. Pengumpulan data bertujuan untuk mengetahui permasalahan dan kebutuhan informasi mengenai sistem penerimaan santri baru pada Pondok Pesantren Bilal Bin Rabah Sukoharjo berbasis *web*.

5) Observasi Aplikasi

Observasi aplikasi bertujuan untuk mencari gambaran sistem yang akan dibangun dengan melihat beberapa aplikasi yang sudah ada. Observasi dilakukan dengan melihat modul yang dibutuhkan untuk sistem seperti formulir pendaftaran, pembayaran, cetak kartu, dan laporan pendaftaran.

#### 6) Analisis dan Perancangan Sistem

Analisis dan perancangan sistem penerimaan santri baru berbasis *web* pada Pondok Pesantren Bilal Bin Rabah bertujuan untuk mengetahui kesiapan data dan rancangan dari sistem penerimaan santri baru yang diharapkan dapat memberikan suatu solusi kepada Pondok Pesantren Bilal Bin Rabah. Perancangan sistem pada aplikasi ini menggunakan *UML (Unified Modeling Language)*.

#### 7) Implementasi dan Pengembangan Sistem

Membuat rancang bangun sistem penerimaan santri baru berbasis *web* menggunakan bahasa pemrograman *PHP*, *database MySQL* sebagai media penyimpanan datanya, dan *CodeIgniter 4* sebagai *framework*-nya.

#### 8) Pengujian Sistem

Pengujian sistem merupakan tahap setelah sistem berhasil dibuat. Pengujian sistem bertujuan untuk mengetahui kelebihan, kelemahan serta mengetahui kelayakan suatu sistem untuk digunakan.

#### 9) Penerapan Sistem

Penerapan sistem merupakan tahap setelah sistem berhasil diuji tanpa ada kesalahan pada sistem yang telah dibuat. Setelah berhasil maka sistem akan diterapkan untuk membantu sistem penerimaan santri baru pada Pondok Pesantren Bilal Bin Rabah Sukoharjo.

### **2.3. Teori Pendukung**

#### **2.3.1. Pengertian Penerimaan Peserta Didik Baru**

Penerimaan peserta didik baru pada dasarnya untuk memperlancar dan mempermudah dalam proses pendaftaran siswa-siswi baru, pendataan dan pembagian kelas seorang siswa-siswi. Sehingga dapat terorganisir, teratur dengan cepat dan tepat dengan beberapa persyaratan yang telah ditentukan oleh sekolah. Proses penerimaan peserta didik baru merupakan salah satu kewajiban pihak sekolah dan Dinas Pendidikan setiap tahun ajaran baru. Pada proses PPDB biasanya terdapat proses seleksi administrasi dan akademis calon siswa untuk memasuki jenjang pendidikan setingkat lebih tinggi (Pasaribu, 2017).

Penerimaan siswa baru merupakan gerbang awal yang harus dilalui peserta didik dan sekolah didalam penyaringan objek-objek pendidikan. Peristiwa penting bagi suatu sekolah, karena peristiwa ini merupakan titik awal yang menentukan kelancaran tugas suatu sekolah. Kesalahan dalam penerimaan siswa baru dapat menentukan sukses tidaknya usaha pendidikan disekolah yang bersangkutan. Penerimaan siswa baru dilakukan bukanlah suatu hal yang ringan. Sekolah harus menyiapkan strategi-strategi yang tepat dalam menjalankannya, supaya dapat menarik siswa-siswa berkualitas yang mana *input* sekolah juga bisa lebih baik sehingga proses belajar bisa maksimal dan kualitas sekolah meningkat (Sarwindah, 2018).

### 2.3.2. XAMPP

*XAMPP* merupakan perangkat lunak bebas, yang mendukung banyak sistem operasi, dan merupakan kompilasi dari beberapa program. Seperti *Apache*, *MYSQL*, *PHPP*, dan *Perl*. *XAMPP* adalah *tool* yang menyediakan paket perangkat lunak dalam satu buah paket. Dalam paket *XAMPP* sudah terdapat *Apache (Web Server)*, *Mysql (Database)*, *PHP (server side scripting)*, *Perl*, *FTP server*, *PhpMyAdmin*, dan berbagai pustaka lainnya (Novendri, dkk. 2019).

*MYSQL* disebut juga *SQL* yang merupakan singkatan dari *Structured Query Language*. *SQL* merupakan bahasa terstruktur yang khusus digunakan untuk mengolah *database*. *MYSQL* adalah sebuah sistem pengelolaan *database* yang bersifat *open source*. *MYSQL* merupakan sistem pengelolaan *database* yang bersifat *relational*. Artinya, data yang dikelola dalam *database* yang akan diletakkan pada beberapa tabel yang terpisah sehingga manipulasi data akan jauh lebih cepat. *MYSQL* dapat digunakan untuk mengelola *database* mulai dari yang kecil sampai dengan yang sangat besar (Novendri, dkk. 2019).

*SQL* dapat diartikan sebagai antar muka standar untuk sistem manajemen relasional, termasuk sistem yang beroperasi pada komputer pribadi. *SQL* memungkinkan seorang pengguna untuk mengetahui dimana lokasinya, atau bagaimana informasi tersebut disusun. *SQL* lebih mudah digunakan dibandingkan dengan bahasa pemrograman, tetapi rumit dibandingkan *software* lembar kerja dan

pengolah data. *SQL* juga merupakan bahasa pemrograman yang dirancang khusus untuk mengirimkan suatu perintah *query* (pengaksesan data berdasarkan pengalamatan tertentu) terhadap sebuah *database*. Kebanyakan *software* database mengimplementasikan *SQL* secara sedikit berbeda, tapi seluruh *database SQL* mendukung subset standar yang ada. Dengan kata lain, *SQL* adalah perintah atau bahasa yang melekat dalam *SMBD*. Sebagai suatu bahasa permintaan, *SQL* didukung oleh *SMBD*, seperti *MySQL Server*, *MySQL*, *PostgreSQL*, *Interbase*, dan *Oracle*. Selain itu *SQL* juga didukung oleh *database* bukan *server*, seperti *MS Access* maupun *Paradox* (Novendri, dkk. 2019).

*Database* atau basis data adalah kumpulan terorganisasi dari data-data yang berhubungan sedemikian rupa sehingga mudah disimpan, dimanipulasi serta dipanggil oleh pengguna. Terminologi hubungan berarti data mendeskripsikan domain tertentu sehingga pengguna mudah untuk mendapatkan jawaban atas pertanyaan yang diajukan ke basis data tersebut. Sedangkan pengertian sistem basis data adalah sebagai koleksi dari data-data yang terorganisasi sehingga data mudah disimpan dan dimanipulasi (Novendri, dkk. 2019).

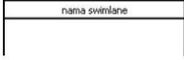
### **2.3.3. Unified Modelling Language (UML)**

UML (Unified Modeling Language) adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan requirement, membuat analisis & desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek (Rosa dan Shalahuddin, 2016). Ada beberapa macam diagram dalam Unified Modeling Language (UML) sebagai berikut:

#### 1) *Activity Diagram*

Menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan adalah diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan *actor*, jadi aktivitas yang dapat dilakukan oleh sistem (Rosa dan Shalahuddin, 2016). Simbol-simbol yang digunakan pada *Activity diagram* dapat dilihat pada Tabel 2.1.

Tabel 2.1. Simbol *Activity Diagram*

GAMBAR	NAMA	KETERANGAN
	Status awal	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal
	Aktivitas	Aktivitas yang dilakukan sistem, aktivitas biasanya dilakukan dengan kata kerja.
	Percabangan	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
	Penggabungan	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
	Status akhir	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
	<i>Swimlane</i>	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

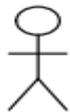
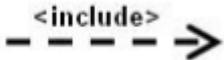
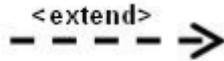
## 2) *Use Case Diagram*

*Use case* atau diagram *use case* merupakan pemodelan untuk kelakuan (behavior) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case* (Rosa dan Shalahuddin, 2016).

- a) Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
- b) *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

Pada *Use Case Diagram* terdapat simbol-simbol yang digunakan dalam permodelan menggunakan *UML*. Simbol-simbol yang digunakan pada *Use Case diagram* dapat dilihat pada Tabel 2.2.

Tabel 2.2. Simbol *Use Case Diagram*

GAMBAR	NAMA	KETERANGAN
	Aktor	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri.
	<i>Generalisasi</i>	Hubungan dimana objek anak ( <i>descendent</i> ) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk ( <i>ancestor</i> ).
	<i>Include</i>	Menspesifikasikan bahwa <i>use case</i> sumber secara <i>eksplisit</i> .
	<i>Extend</i>	Menspesifikasikan bahwa <i>use case</i> target memperluas perilaku dari <i>use case</i> sumber pada suatu titik yang diberikan.
	<i>Association</i>	Asosiasi, komunikasi antara aktor dan <i>use case</i> yang berpartisipasi pada <i>use case</i> atau <i>use case</i> memiliki interaksi dengan aktor.
	<i>Use Case</i>	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja diawal-awal frase nama <i>use case</i> .

### 3) *Class Diagram*

*Class diagram* merupakan diagram yang selalu ada di permodelan sistem berorientasi objek. *Class diagram* menunjukkan hubungan antar class dalam sistem yang sedang dibangun dan bagaimana mereka saling berkolaborasi untuk mencapai suatu tujuan. Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem (Rosa dan Shalahuddin, 2016). Kelas memiliki apa yang disebut atribut dan metode atau operasi sebagai berikut:

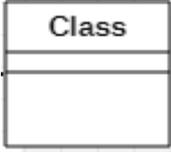
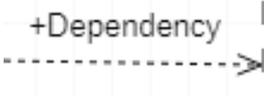
- a) Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas.
- b) Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut:

- a) Kelas main Kelas yang memiliki fungsi awal dieksekusi ketika sistem dijalankan.
- b) Kelas yang menangani tampilan sistem (*view*) Kelas yang mendefinisikan dan mengatur tampilan ke pemakai.
- c) Kelas yang diambil dari pendefinisian *use case (controller)* Kelas yang menangani fungsi-fungsi yang harus ada diambil dari pendefinisian *use case*, kelas ini biasanya disebut dengan kelas proses yang menangani proses bisnis pada perangkat lunak.
- d) Kelas yang diambil dari pendefinisian data (model) Kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data.

Pada *Class Diagram* terdapat simbol-simbol yang digunakan dalam permodelan menggunakan *UML*. Simbol-simbol yang digunakan pada *Class Diagram* dapat dilihat pada Tabel 2.3.

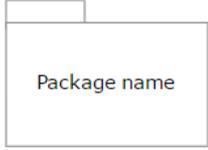
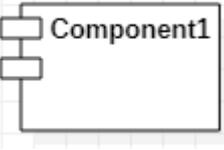
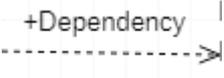
Tabel 2.3. Simbol *Class Diagram*

GAMBAR	NAMA	KETERANGAN
	Kelas	Operasi kelas pada struktur sistem.
	<i>Interface</i>	Sama dengan konsep <i>interface</i> dalam pemrograman berorientasi objek.
	<i>Association</i>	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan <i>multiplicity</i> .
	<i>Directed Association</i>	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan <i>multiplicity</i> .
	<i>Generalisasi</i>	Relasi antar kelas dengan makna generalisasi spesialisasi (umum khusus)
	<i>Depedency</i>	Relasi antar kelas dengan makna kebergantungan antar kelas
	<i>Aggregation</i>	Relasi antar kelas dengan makna semua bagian ( <i>whole-part</i> ).

#### 4) *Component Diagram*

Diagram komponen atau *component diagram* dibuat untuk menunjukkan organisasi dan ketergantungan diantara kumpulan komponen dalam sebuah sistem. Diagram komponen fokus pada komponen sistem yang dibutuhkan dan ada di dalam sistem (Rosa dan Shalahuddin, 2016). Simbol-simbol yang digunakan pada *Component Diagram* dapat dilihat pada Tabel 2.4.

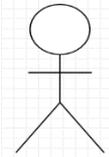
Tabel 2.4. Simbol *Component Diagram*

GAMBAR	NAMA	KETERANGAN
	<i>Package</i>	<i>Package</i> merupakan sebuah bungkus dari satu atau lebih komponen.
	Komponen	Komponen sistem.
	<i>Dependency</i>	Kebergantungan antar komponen, arah panah mengarah pada komponen yang dipakai.
	<i>Interface</i>	Sama dengan konsep <i>interface</i> dalam pemrograman berorientasi objek.
	<i>Link</i>	Relasi antar komponen.

### 5) *Sequence Diagram*

*Sequence diagram* atau diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirim dan diterima antar objek. Banyaknya diagram sekuensial yang harus digambar adalah minimal sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksinya pesan sudah dicakup pada diagram sekuensial sehingga semakin banyak *use case* yang didefinisikan maka diagram sekuensial yang harus dibuat juga semakin banyak (Rosa dan Shalahuddin, 2016). Simbol-simbol yang digunakan pada *Sequence Diagram* dapat dilihat pada Tabel 2.5.

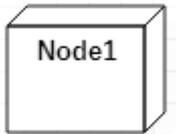
Tabel 2.5. *Sequence Diagram*

GAMBAR	NAMA	DESKRIPSI
	Aktor/ <i>Actor</i>	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat sendiri.
	<i>Lifeline</i>	Menggambarkan sebuah objek dalam sebuah sistem atau salah satu komponennya.
	Pesan tipe <i>create</i>	Pembuatan sebuah <i>message</i> sederhana antar elemen dan juga mengindikasikan komunikasi antara objek.
	Pesan tipe <i>call</i>	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri.
	Pesan tipe <i>send</i>	Menyatakan bahwa suatu objek mengirimkan data / masukan / informasi ke objek lainnya.
	Pesan tipe <i>return</i>	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu.
	Pesan tipe <i>destroy</i>	Menyatakan suatu objek mengakhiri hidup objek yang lain.
	Waktu aktif	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini ada sebuah tahapan didalamnya.
	objek	Menyatakan suatu objek membuat objek lain.

#### 6) *Deployment Diagram*

Diagram ini bersifat statis. Diagram ini memperlihatkan konfigurasi saat aplikasi dijalankan (saat *run time*). Dengan ini memuat simpul–simpul (*node*) beserta komponen–komponen yang ada di dalamnya. *Deployment diagram* berhubungan erat dengan diagram kompoen dimana *deployment diagram* memuat satu atau lebih komponen–komponen. Diagram ini sangat berguna saat aplikasi berlaku sebagai aplikasi yang dijalankan pada banyak mesin (*distributed computing*) (Rosa dan Shalahuddin, 2016). Simbol–simbol yang digunakan pada *Deployment Diagram* dapat dilihat pada Tabel 2.6.

Tabel 2.6. *Deployment Diagram*

GAMBAR	NAMA	KETERANGAN
	<i>Package</i>	Package merupakan sebuah bungkusan dari satu atau lebih <i>node</i> .
	<i>Node</i>	Menggambarkan perangkat keras ( <i>hardware</i> ) atau perangkat lunak ( <i>software</i> ).
	<i>Link</i>	Menggambarkan hubungan antara <i>node</i> .
	<i>Dependency</i>	Kebergantungan antar <i>node</i> , arah panah mengarah pada <i>node</i> yang dipakai.

#### 2.3.4. *PHP*

Bahasa pemrograman *PHP* merupakan bahasa pemrograman untuk membuat *website* yang bersifat *server-side scripting*. Karena pemrosesan program *PHP* dilakukan didalam lingkungan *web browser*, *PHP* dikatakan sebagai bahasa sisi *server (server-side)* Selain *Apache*, *PHP* juga mendukung beberapa *web server* lain, seperti *Microsoft ISS*, *Caudium*, dan *PWS*. *PHP* dapat memanfaatkan database

untuk menghasilkan halaman *web* yang dinamis. Sistem manajemen *database* yang sering digunakan bersama *PHP* adalah *MYSQL*. Namun, *PHP* juga mendukung sistem manajemen *Database Oracle, Microsoft Access, Interbase, d-Base, dan PostgreSQL* (Novendri, dkk. 2019).

### **2.3.5. HTML**

*HTML* atau *Hyper Text Markup Language* merupakan sebuah bahasa pemrograman terstruktur yang dikembangkan untuk membuat halaman *website* yang dapat diakses atau ditampilkan menggunakan *Web Browser*. *HTML* sendiri secara resmi lahir pada tahun 1989 oleh Tim Berners Lee dan dikembangkan oleh *World Wide Web Consortium (W3C)*, yang kemudian pada tahun 2004 dibentuklah *Web Hypertext Application Tecnology Working Group (WHATG)* yang bertanggung jawab akan perkembangan bahasa *HTML*. *HTML 5* adalah sebuah versi yang mendukung tidak hanya gambar dan teks, namun juga menu interaktif, audio, video, dan lain sebagainya (Setiawan, 2017).

### **2.3.6. CodeIgniter 4**

*CodeIgniter* adalah Kerangka Pengembangan Aplikasi untuk membangun situs *web* menggunakan *PHP*. Tujuan *framework* ini adalah untuk dapat mengembangkan proyek lebih cepat daripada yang menulis kode dari awal, dengan menyediakan kumpulan pustaka yang kaya untuk tugas-tugas yang umumnya dibutuhkan, serta antarmuka sederhana dan struktur logis untuk mengakses pustaka ini. *CodeIgniter* memungkinkan pengguna secara kreatif fokus pada proyek yang dibuat dengan meminimalkan jumlah kode yang dibutuhkan untuk tugas tertentu (CodeIgniter, 2021).

### **2.3.7. Pengukuran Kualitas Website**

*Website* atau *Web* adalah sebuah koleksi keterhubungan dokumen-dokumen yang disimpan di-*internet* dan diakses menggunakan *protocol (Hyper Text Transfer Protocol)*. Intinya bahwa penggunaan *internet* bisa memanfaatkan berbagai macam fasilitas dengan biaya murah tanpa harus datang secara langsung

ketempatnya. Informasi atau dokumen yang dapat diakses dapat berupa data teks, gambar, atau animasi, video, suara atau kombinasi diantaranya dan bahkan komunikasi bisa dilakukan secara langsung dengan suara dan video baik yang bersifat statis maupun dinamis yang membentuk satu rangkaian bangunan yang saling terkait, dimana masing-masing dihubungkan dengan jaringan-jaringan halaman (Sarwindah, 2018).

Pengukuran kualitas *website* dengan menggunakan metode *WebQual* terdiri dari enam dimensi, yaitu aksesibilitas, interaksi, kecukupan interaksi, kegunaan dari konten, gaya hidup, dan kepribadian. Perilaku pengguna *internet* mempengaruhi persepsi kualitas layanan dan dimensi aksesibilitas, interaksi, kecukupan informasi, kegunaan dari konten, dan gaya hidup (Monalisa dan Rizky, 2021).

#### **2.3.8. *WebQual***

*WebQual* berdasar pada konsep *Quality Function Deployment (QFD)* adalah suatu pengukuran untuk mengukur kualitas dari sebuah *website* berdasarkan instrument yang dapat dikategorikan kedalam tiga variable yaitu: *usability*, *information quality*, dan *interaction quality*. Pada ketiga variabel bertujuan untuk mengukur kepuasan konsumen atau pengguna terhadap kualitas *website*. Metode penelitian ini mulai dikembangkan sejak tahun 1998, Metode ini juga mengalami perubahan dalam penyusunan dimensi dan butir pertanyaan. Versi terbaru *WebQual* yaitu 4.0, *WebQual 4.0* disusun berdasarkan tiga dimensi, yaitu: *usability*, kualitas informasi (*information quality*), dan kualitas interaksi (*interaction quality*) (Monalisa dan Rizky, 2021).