BAB II
LANDASAN TEORI

2.1. Kajian Pustaka

Penelitian oleh (Putra, et al., 2020) menggunakan metode MOORA untuk melakukan penilaian kinerja pegawai pada PDAM Martapura Oku dengan 4 kriteria penilaian yaitu sikap dan perilaku, kemampuan dan keterampilan, kerjasama, tanggung jawab. Permasalahan yang dihadapi dalam penilaian kinerja pegawai di PDAM Martapura dilakukan tanpa alat bantu seperti sistem perhitungan dan hanya menjumlahkan skor untuk mendapatkan nilai yang akhirnya penilaian kinerja pegawai itu tidak berjalan dengan baik sehingga perankingan penilaian kinerja pegawai dirasa terlalu memakan waktu dan kurang efisien, hasil dari perankingan juga akhirnya tidak objektif. Sistem yang dihasilkan mampu melakukan penilaian pegawai dan perangkingan kinerja dari nilai tertinggi sampai dengan nilai terendah.

Penelitian lainnya oleh (Alatas, et al., 2021) menggunakan metode MOORA untuk kenaikan jabatan pegawai dengan kriteria kedisiplinan, pengalaman kerja, alamat, pendidikan, status, penampilan dan usia. Permasalahan yang dihadapi PT. Supra Visual Mandiri saat ini yaitu proses penilaian kinerja pegawainya masih dilakukan secara manual dan tidak transparan sehingga terjadi demotivasi pegawai dan pencapaian sasaran perusahaan menjadi tidak optimal. Sistem yang dihasilkan mampu melakukan seleksi penilaian kinerja akan lebih efektif, transparan dan tepat karena berdasarkan nilai kriteria dan bobot yang sudah ditentukan.

Penelitian oleh (Nainggolan, et al., 2022) menggunakan metode MOORA untuk menilai indeks kinerja sales marketing pada PT. ALFA SCORPII dengan kriteria empati, disiplin, kualitas kerja, kuantitas kerja, kerjasama, komitmen, pengendalian sikap, integritas, komunikasi, pelayanan, inisiatif. Sistem pendukung keputusan penilaian indeks kinerja *sales marketing* dapat membantu manajemen terkait dalam menentukan nilai kinerja dari masing

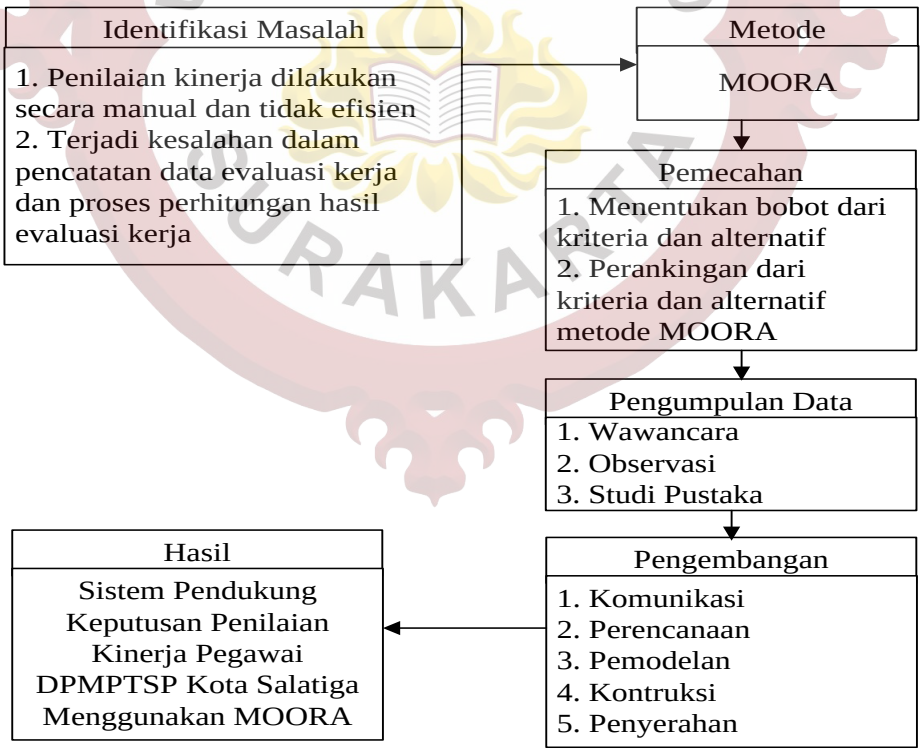
masing *sales marketing*. Penerapan metode MOORA dapat memberikan dengan hasil perhitungan dengan spesifik terhadap data tiap alternatif.

Penelitian sebelumnya oleh (Hasibuan, et al., 2019) melakukan perbandingan metode MOORA dan TOPSIS dengan hasil rata-rata presentase sensitivitas metode MOORA (-1,61%) lebih baik daripada metode TOPSIS (-7,96%).

Penelitian oleh (Afrisawati & Sahren, 2020) juga melakukan perbandingan metode MOORA dan WASPAS dengan hasil metode MOORA lebih efektif digunakan karena menghasilkan nilai alternatif yang lebih cepat, tepat dan mudah

2.2. Kerangka Pemikiran

Kerangka pemikiran pemecahan masalah penelitian ini digambarkan pada gambar 2.1.



Gambar 2.1. Kerangka Pemikiran

Penelitian ini diawali dengan munculnya permasalahan yaitu penilaian kinerja pegawai yang masih dilakukan secara manual dan belum efektifnya dalam

penilaian kinerja pegawai pada DPMPTSP Kota Salatiga. Metode penilaian menggunakan metode MOORA. Perlu dilakukan penentuan bobot untuk kriteria yang akan ditentukan lalu merankingnya. Pengumpulan data yang dipergunakan yaitu wawancara, observasi dan studi pustaka. Selanjutnya melakukan pengembangan, pengembangan terbagi menjadi lima tahapan yaitu komunikasi, perencanaan, pemodelan, kontruksi kemudian dilakukan penyerahan ke DPMPTSP Kota Salatiga.

2.3. Teori Pendukung

2.3.1. Sistem Pendukung Keputusan

Konsep sistem pendukung keputusan pertama kali diperkenalkan pada tahun 1970-an oleh Michael S.Scott Marton dengan istilah *management decision system*. Konsep sistem pendukung keputusan ditandai dengan sistem interaktif berbasis komputer yang membantu mengambil keputusan memanfaatkan data dan model untuk menyelesaikan masalah yang tidak terstruktur (Kusumadewi, 2018).

Konsep SPK merupakan sebuah sistem interaktif berbasis komputer yang membantu pembuatan keputusan memanfaatkan data dan model untuk menyelesaikan masalah-masalah yang bersifat tidak terstruktur dan semi terstruktur (Marbun & Sinaga, 2019). SPK dirancang untuk menunjang seluruh tahapan pembuatan keputusan, yang dimulai dari tahapan mengidentifikasi masalah, memilih data yang relevan, menentukan pendekatan yang digunakan dalam proses pembuatan keputusan sampai pada kegiatan mengevaluasi pemilihan alternatif (Limbong, et al., 2020).

Pada dasarnya sistem pendukung keputusan dirancang untuk mendukung seluruh tahap pengambilan keputusan mulai dari mengidentifikasi masalah, memilih data yang relevan, menentukan pendekatan yang digunakan dalam pengambilan keputusan sampai mengevaluasi pemilihan alternatif (Sarwandi, et al., 2023).

Sistem pendukung keputusan adalah suatu sistem informasi spesifik yang ditujukan untuk membantu manajemen dalam mengambil keputusan yang berkaitan dengan persoalan yang bersifat (Kusumadewi, 2018):

1. Terstruktur, yaitu berhubungan dengan persoalan yang telah diketahui sebelumnya dengan penyelesaian standar aturan yang telah ditentukan.
2. Semi terstruktur, yaitu berhubungan dengan persoalan yang belum diketahui sebelumnya, dengan parameter yang sudah ada.
3. Tidak terstruktur, yaitu berhubungan dengan persoalan baru yang cukup pelik, karena banyaknya data yang belum diketahui.

Sistem pendukung keputusan terdiri dari 3 komponen utama atau subsistem yaitu (Kusumadewi, 2018):

1. Subsistem Data (*Database*)

Subsistem data merupakan komponen sistem pendukung keputusan penyedia data bagi sistem. Data yang dimaksud disimpan dalam suatu pangkalan data (*database*) yang diorganisasikan oleh suatu sistem yang disebut dengan sistem manajemen pangkalan data (*Data Base Management System* atau DBMS). Melalui DBMS inilah data dapat diambil dan dievakuasi dengan cepat. Pangkalan data dalam sistem pendukung keputusan berasal dari dua sumber yaitu sumber internal (dari dalam perusahaan) dan sumber eksternal (dari luar perusahaan). Data eksternal ini sangat berguna bagi manajemen dalam mengambil keputusan.

2. Subsistem Model (*Model Base*)

Keunikan sistem pendukung keputusan adalah kemampuannya dalam mengintegrasikan data dengan model-model keputusan. Model adalah suatu peniruan dari alam nyata. Kendala yang sering dihadapi dalam merancang suatu model adalah bahwa model yang disusun ternyata tidak mampu mencerminkan seluruh variabel alam nyata. Sehingga keputusan yang diambil yang didasarkan pada model tersebut menjadi tidak akurat dan tidak sesuai dengan kebutuhan. Oleh karena itu dalam menyimpan berbagai model pada sistem pangkalan model harus tetap dijaga fleksibilitasnya, artinya harus ada fasilitas yang mampu membantu pengguna untuk memodifikasi atau menyempurnakan model seiring dengan perkembangan pengetahuan.

3. Subsistem Dialog (*User System Interface*)

Keunikan lain dari sistem pendukung keputusan adalah adanya fasilitas yang mampu mengintegrasikan sistem yang terpasang dengan pengguna secara

interaktif. Fasilitas atau subsistem ini dikenal sebagai subsistem dialog, inilah sistem diartikulasikan dan diimplementasikan sehingga pengguna atau pemakai dapat berkomunikasi dengan sistem yang dirancang.

2.3.2. Penilaian Kinerja

Penilaian kinerja adalah proses untuk mengukur prestasi kerja pegawai berdasarkan peraturan yang telah ditetapkan, dengan cara membandingkan sasaran (hasil kerjanya) dengan standar pekerjaan yang telah ditetapkan selama periode tertentu. Standar kerja tersebut dapat dibuat baik secara kualitatif maupun kuantitatif. Penilaian prestasi kerja adalah suatu pendekatan dalam melakukan penilaian prestasi kerja para pegawai yang di dalamnya terdapat berbagai faktor seperti (Dessler, 2016):

1. Penilaian dilakukan pada manusia sehingga disamping memiliki kemampuan tertentu juga tidak luput dari berbagai kelemahan dan kekurangan.
2. Penilaian yang dilakukan pada serangkaian tolak ukur tertentu yang realistik, berkaitan langsung dengan tugas seseorang serta kriteria yang ditetapkan dan diterapkan secara obyektif.
3. Hasil penilaian harus disampaikan kepada pegawai yang dinilai.

Penilaian kinerja yang baik adalah yang mampu untuk menciptakan gambaran yang tepat mengenai kinerja pegawai yang dinilai. Penilaian tidak hanya ditujukan untuk menilai dan memperbaiki kinerja yang buruk, namun juga untuk mendorong para pegawai untuk bekerja lebih baik lagi. Berkaitan dengan hal ini, penilaian kinerja membutuhkan standar pengukuran, cara penilaian dan analisa data hasil pengukuran, serta tindak lanjut atas hasil pengukuran. Elemen-elemen utama dalam sistem penilaian kinerja adalah (Dessler, 2016):

1. *Validity*

Berarti keabsahan standar tersebut sesuai dengan jenis pekerjaan yang dinilai. Keabsahan yang dimaksud di sini adalah standar tersebut memang benar-benar sesuai atau relevan dengan jenis pekerjaan yang akan dinilai tersebut.

2. *Agreement*

Berarti persetujuan, yaitu standar penilaian tersebut disetujui dan diterima oleh semua pegawai yang akan mendapat penilaian. Ini berkaitan dengan prinsip *validity* di atas.

3. *Realism*

Berarti standar penilaian tersebut bersifat realistis, dapat dicapai oleh para pegawai dan sesuai dengan kemampuan pegawai.

4. *Objectivity*

Berarti standar tersebut bersifat obyektif, yaitu adil, mampu mencerminkan keadaan yang sebenarnya tanpa menambah atau mengurangi kenyataan dan sulit untuk dipengaruhi oleh bias-bias penilai.

2.3.3. MOORA

MOORA diperkenalkan oleh Brauers pada tahun 2004 sebagai *multi-objective optimization,* mengoptimalkan beberapa fungsi perhitungan nilai yang lebih dari satu agar memiliki nilai efektif yang dapat dicapai dalam batasan-batasan wilayah tersendiri agar yang dicari tidak melebar kemana-mana, yang dapat digunakan untuk memecahkan berbagai masalah dalam pengambilan keputusan.

Metode MOORA memiliki tingkat selektifitas yang baik dalam memutuskan suatu alternatif. Pendekatan yang dilakukan MOORA diartikan sebagai suatu proses secara bersamaan guna mengoptimalkan dua atau lebih kriteria yang saling bertentangan pada beberapa masalah. Selain itu metode ini juga memperoleh hasil yang lebih akurat dan tepat sasaran dalam membantu pangambilan keputusan. Bila dibandingkan dengan metode yang lain metode MOORA bahkan lebih sederhana dan mudah diimplementasikan (Astuti, 2020). Langkah penyelesaian dari Metode MOORA yaitu sebagai berikut:

1. Menginputkan Nilai Kriteria

Menentukan tujuan, mengidentifikasi dan mengevaluasi atribut, dan menginputkan kriteria pada suatu alternatif.

2. Membuat Matrik Keputusan

$$X_{ij} = \begin{bmatrix} X_{11} & X_{12} & \dots & X_{1n} \\ X_{21} & X_{22} & \dots & X_{3n} \\ & \dots & . & \\ X_{m1} & X_{m2} & \dots & X_{m3} \end{bmatrix}$$

3. Matriks Normalisasi

Bertujuan untuk menyatukan setiap elemen matriks sehingga memiliki nilai yang seragam. Braures menyimpulkan untuk penyebut, pilihan terbaik adalah akar kuadrat dari jumlah kuadrat dan setiap alternatif per atribut.

$$X_{ij} = \frac{x_{ij}}{\sqrt{\sum_{j=1}^{m} x_{ij}^2}}$$

4. Menentukan Matriks Normalisasi Terbobot

Untuk optimasi Multi-Objektif, kinerja yang dinormalisasi ditambahkan dalam kasus maksimasi (untuk atribut yang menguntungkan) dan dikurangi dalam hal minimasi (untuk atribut non menguntungkan). Namun saat atribut bobot dimasukkan, maka dapat dirumuskan sebagai berikut :

$$Yi = \sum_{j=1}^{g} W\,jXij - \sum_{j=g+1}^{g} W\,jXij$$

2.3.4. *Website*

Website merupakan kumpulan halaman-halaman yang digunakan untuk menampilkan informasi teks, gambar diam atau gerak, animasi, suara dan atau gabungan dari semuanya, baik yang bersifat statis maupun dinamis yang membentuk satu rangkaian bangunan yang saling terkait, yang masing-masing dihubungkan dengan jaringan-jaringan halaman (Sibero, 2014). *Website* dibagi menjadi 2 yaitu:

1. *Website* statis

Website statis adalah *website* yang mempunyai halaman konten yang tidak berubah-ubah.

2. *Website* dinamis

Website dinamis merupakan *website* yang secara struktur ditujukan untuk *update* sesering mungkin.

2.3.5. *Hosting*

Hosting dapat diartikan sebagai ruangan yang terdapat dalam harddisk tempat menyimpan berbagai data, file-file, gambar dan lain sebagainya yang akan ditampilkan di website. Besarnya data yang bisa dimasukkan tergantung dari besarnya *hosting* yang disewa/dipunyai, semakin besar *hosting* semakin besar pula data yang dapat dimasukkan dan ditampilkan dalam website. *Hosting* juga diperoleh dengan menyewa. Besarnya *hosting* ditentukan ruangan *harddisk* dengan ukuran MB (*Mega Byte*) atau GB (*Giga Byte*). Lama penyewaan *hosting* rata-rata dihitung per tahun. Penyewaan *hosting* dilakukan dari perusahaan-perusahaan penyewa *hosting* yang banyak dijumpai baik di Indonesia maupun luar negeri (Sibero, 2014).

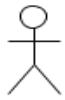
2.3.6. UML (*Unified Modelling Language)*

Unified Modelling Language (UML) adalah sebuah bahasa yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan *class* dan *operation* dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasa-bahasa berorientasi objek seperti C++, Java, C# atau VB.NET. Walaupun demikian, UML tetap dapat digunakan untuk modeling aplikasi prosedural dalam VB atau C (Muslihudin, 2017).

1. *Use Case*

Use case adalah bagian dari tingkat tinggi fungsionalitas yang disebut oleh sistem. Dengan kata lain *use case* menggambarkan bagaimana seseorang menggunakan sistem. (Munawar, 2018). Dibawah ini merupakan simbol-simbol yang digunakan dalam pembuatan *use case*.

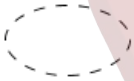
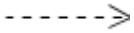
Tabel 2.1. Simbol *Use Case*

Simbol	Nama	Keterangan
	*Actor*	Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
	Dependency	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*).
	*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
	*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*.
	Extend	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
	Association	Apa yang menghubungkan antara objek satu dengan objek lainnya.
	*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
	*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
	*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
	*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi

2. *Class Diagram*

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class diagram* menggambarkan struktur dan deskripsi *class*, *package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain. (Munawar, 2018).

Tabel 2.2. Simbol *Class Diagram*

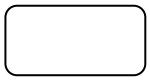
SIMBOL	NAMA	KETERANGAN
	Generalization	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
	*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
	*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
	*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
	Realization	Operasi yang benar-benar dilakukan oleh suatu objek.
	*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri
	Association	Apa yang menghubungkan antara objek satu dengan objek lainnya

3. *Activity Diagram*

Activity diagram menggambarkan alur kerja sebuah proses bisnis dan urutan aktifitas dalam suatu proses. Diagram ini sangat mirip dengan sebuah *flowchart* karena dapat memodelkan sebuah alur kerja dari suatu aktifitas ke aktifitas lainnya atau dari satu aktifitas kedalam keadaan sesaat (*state*). *Activity*

diagram bermanfaat untuk memahami proses secara keseluruhan (Munawar, 2018).

Tabel 2.3. Simbol *Activity Diagram*

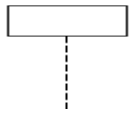
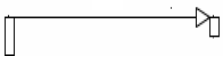
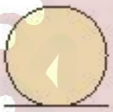
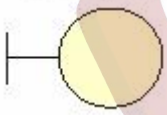
Simbol	Nama	Keterangan
	*State*	Kondisi yang mungkin dialami oleh suatu obyek.
	*Note*	Note digunakan untuk memberikan keterangan ataukomentar
	Aktivitas	Perilaku obyek yang dilakukan saat obyek berada dalam *state* tertentu.
	*Start State*	*Start state* digunakan untuk memulai diagram *statechart.*
	*End State*	End start digunakan untuk mengakhiri diagram.
	*Decision*	*Decision* digunakan sebagai pilihan untuk pengambilan keputusan.
	Penggabungan / *Join*	digunakan untuk split dan join.pada saat diagram akanmembagi 2, bar ini akanditambahkan. dan sebelum diagram digabung menjadi satu, sebagai join.
	Asosiasi *(association)*	Apa yang menghubungkan antara objek satu dengan objek yang lainnya.

4. *Sequence Diagram*

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). *Sequence diagram* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan *output*

tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan (Munawar, 2018).

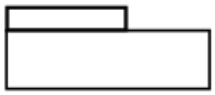
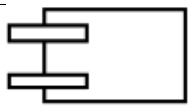
Tabel 2.4. Simbol *Sequence Diagram*

Simbol	Nama	Keterangan
	*LifeLine*	Menggambarkan objek *entity*, antarmuka yang saling berinteraksi.
	*Message*	Menggambarkan pengiriman pesan
	*Actor*	Menggambarkan orang yang sedang berinteraksi dengan sistem
	*Entity Class*	Menggambarkan hubungan kegiatan yang akan dilakukan
	*Boundary Class*	Menggambarkan sebuah penggambaran dari form
	*Control Class*	Menggambarkan penghubung antara *boundary* dengan tabel

5. *Component Diagram*

Component diagram yaitu salah satu jenis diagram pada UML yang menggambarkan *software* pada suatu sistem. *Component diagram* merupakan penerapan *software* dari satu ataupun lebih *class*, dan biasanya berupa file data atau .exe, *source code*, *table*, dokumen dan sebagainya (Munawar, 2018).

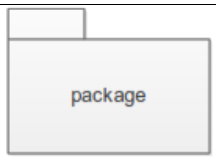
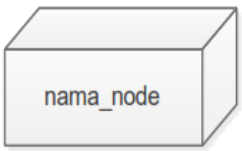
Tabel 2.5. Simbol *Component Diagram*

Simbol	Nama	Keterangan
	*Package*	*Package* merupakan sebuah bungkusan dari satu atau lebih komponen
	*Component*	Komponen sistem
	*Dependency*	Kebergantungan antar komponen, arah panah yang mengarah komponen yang dipakai
	*Interface*	Sama dengan konsep *interface* pada pemrograman berorientasi objek, yaitu sebagai antar muka komponen agar tidak mengakses langsung komponen
	Link	Relasi antar komponen

6. *Deployment Diagram*

Deployment diagram menunjukkan konfigurasi komponen dalam proses eksekusi aplikasi. *Deployment diagram* juga dapat digunakan untuk memodelkan sistem tambahan (*embedded system*) yang menggambarkan rancangan *device*, *node*, dan *hardware*, serta digunakan untuk memodelkan sistem *client/server* (Munawar, 2018).

Tabel 2.6. Simbol *Deployment Diagram*

Simbol	Nama	Keterangan
	*Package*	*Package* merupakan sebuah bungkusan dari satu atau lebih *node*
	*Node*	Biasanya mengacu pada perangkat keras (*hardware*), perangkat lunak yang tidak dibuat sendiri (*software*), jikadidalam node disertakan komponen untuk mengkonsistenkan rancangan maka komponen yang diikutsertakan harus sesuai dengan komponen yang telah didefinisikan sebelumnya pada diagram komponen

Simbol	Nama	Keterangan
	Link	Relasi antar *node*
	*Dependency*	Kebergantungan antar *node*, arah panah mengarah pada *node* yang dipakai.

2.3.7. PHP

PHP adalah bahasa program yang berbentuk *script* yang diletakkan di dalam *server* web. Mulanya PHP diciptakan dari ide Rasmus Lerdof yang membuat *script*. *Script* tersebut sebenarnya dimaksudkan untuk digunakan sebagai program untuk dirinya sendiri. Akan tetapi kemudian dikembangkan lagi menjadi sebuah bahasa yang disebut *Personal Home Page* yang kemudian dikenal dengan nama PHP hingga saat ini (Abdulloh, 2018). Beberapa kelebihan PHP dari bahasa pemrograman *web*, antara lain (Setiawan, 2018):

1. Bahasa pemrograman PHP adalah sebuah bahasa *script* yang tidak melakukan sebuah kompilasi dalam penggunaannya.
2. *Web Server* yang mendukung PHP dapat ditemukan dimana-mana, mulai dari apache, IIS, Lighttpd, hingga Xitami dengan konfigurasi yang relatif mudah.
3. Dalam sisi pengembangan lebih mudah, karena banyaknya milis-milis dan *developer* yang siap membantu dalam pengembangan.
4. Dalam sisi pemahaman, PHP adalah bahasa *scripting* yang paling mudah karena memiliki referensi yang banyak.
5. PHP adalah bahasa *open source* yang dapat digunakan di berbagai mesin (Linux, Unix, Macintosh, Windows) dan dapat dijalankan secara *runtime* melalui *console* serta juga dapat menjalankan perintah-perintah sistem.

2.3.8. MySQL

MySQL (*My Strukture Query Language*) adalah sebuah program pembuat *database* bersifat *open source*, artinya siapa saja boleh menggunakannya dan tidak dicekal. MySQL sebenarnya produk yang berjalan pada *platform* Linux. Karena sifatnya yang *open source*, dapat dijalankan pada semua *platform* baik Windows atau Linux. Selain itu, MySQL juga merupakan program pengakses *database*

yang bersifat jaringan sehingga dapat digunakan untuk aplikasi *multi-user*. Saat ini MySQL telah digunakan hampir semua *programmer database*, terutama dalam pemograman web (Nugroho, 2019). Selain itu MySQL mempunyai beberapa kelebihan dibanding *database* lain, yaitu:

1. MySQL sebagai *Database Management Systems* (DBMS).
2. MySQL sebagai *Relation Database Management Systems* (RDBMS).
3. MySQL adalah sebuah *software database* yang *open source* yang artinya program ini bersifat gratis atau bebas digunakan oleh siapa saja tanpa harus membeli dan membayar lisensi kepada pembuatnya.
4. MySQL merupakan sebuah *database server* sehingga dapat dihubungkan ke media internet yang dapat diakses dari jauh.
5. MySQL merupakan sebuah *database client*. MySQL dapat melakukan *query* yang mengakses *database* pada *server*.
6. MySQL mampu menerima *query* yang bertumpuk dalam satu permintaan atau yang disebut *Multi Threading*.
7. MySQL merupakan *database* yang mampu menyimpan data berkapasitas sangat besar hingga berukuran *gigabyte* sekalipun.
8. MySQL didukung oleh *driver* ODBC sehingga dapat diakses menggunakan aplikasi apa saja termasuk visual seperti Delphi maupun Visual Basic.
9. MySQL adalah *database* menggunakan enkripsi *password*.
10. MySQL merupakan *server database* yang *multi user* yang artinya tidak hanya digunakan oleh satu orang saja akan tetapi dapat digunakan oleh banyak orang.
11. MySQL dapat menciptakan lebih dari 16 kunci per tabel dan dalam satu kunci memungkinkan berisi belasan *field* (kolom).
12. MySQL mendukung *field* yang dijadikan sebagai kunci primer dan kunci unik (*unique*).
13. MySQL didukung oleh sebuah komponen C atau perl API sehingga dapat diakses melalui sebuah program aplikasi yang berada dibawah protokol internet berupa web.

14. MySQL memiliki kecepatan dalam pembuatan tabel maupun pengupdatean tabel.
15. MySQL menggunakan bahasa permintaan yang standar yaitu SQL yaitu sebuah bahasa permintaan yang distandarkan pada beberapa *database server* seperti Oracle, PostGreSQL dan lain-lain.

2.3.9. XAMPP

XAMPP merupakan singkatan dari X (sistem operasi pada komputer), A (Apache), M (MySQL), P (PHP), P (Perl). XAMPP adalah *software* yang bersifat *open source* dan mendukung dari beberapa sistem operasi dan gabungan dari beberapa program. Program yang terkandung dalam XAMPP mendukung dari beberapa bahasa pemrograman seperti HTML, Javascript, CSS, PHP, SQL, dan lain-lain. Dalam XAMPP, sudah terkandung apache, yaitu *localhost* atau *web server* yang dapat digunakan dalam proses pembuatan *website* (Aziz & Tampati, 2015).

2.3.10. Bootstrap

Bootstrap merupakan *framework* ataupun *tools* untuk membuat aplikasi web ataupun situs web *responsive* secara cepat, mudah dan gratis. Bootstrap terdiri dari CSS dan HTML untuk menghasilkan *grid, layout, typography, table, form, navigation,* dan lain-lain. Di dalam Bootstrap juga sudah terdapat *jQuery plugins* untuk menghasilkan komponen UI yang bagus seperti *transitions, modal, dropdown, scrollspy, tooltip, tab, popover, alert, button, carousel* dan lain-lain.

Bootstrap diciptakan oleh dua orang *programmer* di Twitter, yaitu Mark Otto dan Jacob Thornton pada tahun 2011. Pada saat itu para *programmer* di twitter menggunakan berbagai macam *tool* dan *libary* untuk melaksanakan pekerjaan, sehingga tidak ada standarisasi dan akibatnya sulit untuk dikelola sehingga Mark Otto dan Jacob Thornton tergerak untuk menciptakan satu *tool* ataupun *framework* yang dapat digunakan bersama di lingkungan internal twitter. Sejak diluncurkan pada bulan agustus 2011, Bootstrap telah berevolusi dari sebuah proyek yang hanya berbasis CSS menjadi sebuah tool ataupun *framework*

yang lebih lengkap yang juga berisi *javascript plugin, icon, forms* dan *button* (Alatas, 2018).

2.3.11. *Black-box testing*

Black-box testing adalah pengujian yang dilakukan hanya mengamati hasil eksekusi melalui data uji dan memeriksa fungsional dari perangkat lunak. Jadi dianalogikan seperti melihat suatu kotak hitam, hanya bisa melihat penampilan luarnya saja tanpa mengetahui apa sesungguhnya yang terjadi dalam proses detilnya (hanya mengetahui *input* dan *output*). Pengujian *black-box* berfokus pada persyaratan fungsional perangkat lunak. Pengujian ini memungkinkan analis sistem memperoleh kumpulan kondisi *input* yang akan mengerjakan seluruh keperluan fungsional program. Tujuan metode ini mencari kesalahan pada (Pressman, 2015):

1. Fungsi yang salah atau hilang.
2. Kesalahan pada *interface*.
3. Kesalahan pada struktur data atau akses *database*.
4. Kesalahan performansi.
5. Kesalahan inisialisasi dan tujuan akhir.

Ujicoba *black-box* diaplikasikan di beberapa tahapan berikutnya. Karena ujicoba *black-box* dengan sengaja mengabaikan struktur kontrol, sehingga perhatiannya difokuskan pada informasi domain. Ujicoba didesain untuk dapat menjawab pertanyaan- pertanyaan berikut (Pressman, 2015):

1. Bagaimana validitas fungsionalnya diuji?
2. Jenis input seperti apa yang akan menghasilkan kasus uji yang baik ?
3. Apakah sistem secara khusus sensitif terhadap nilai input tertentu ?
4. Bagaimana batasan-batasan kelas data diisolasi?
5. Berapa rasio data dan jumlah data yang dapat ditoleransi oleh sistem?
6. Apa akibat yang akan timbul dari kombinasi spesifik data pada operasi sistem?

Dengan mengaplikasikan ujicoba *black-box*, diharapkan dapat menghasilkan sekumpulan kasus uji yang memenuhi kriteria berikut (Pressman, 2015):

1. Kasus uji yang berkurang, jika jumlahnya lebih dari 1, maka jumlah dari uji kasus tambahan harus didesain untuk mencapai ujicoba yang cukup beralasan.
2. Kasus uji yang memberitahukan sesuatu tentang keberadaan atau tidaknya suatu jenis kesalahan, daripada kesalahan yang terhubung hanya dengan suatu ujicoba yang spesifik.

2.3.12. Pengujian Beta

Pengujian *beta* merupakan metode untuk memeriksa dan mengesahkan suatu aplikasi. Pengujian *beta* digunakan untuk menggambarkan proses pengujian *eksternal* dimana aplikasi dapat diedarkan kepada orang lain seperti *user* yang berpotensi menggunakan aplikasi untuk kehidupan sehari-hari. Tujuan dari pengujian *beta* dapat beraneka ragam seperti kesempatan media pers untuk menuliskan masukkan dari *user* untuk mengatasi *bugs* dan kesalahan yang ada. Pengujian *beta* dilakukan dengan menyebar kuisioner kepada pengguna sistem (Pressman, 2015).

Kuisioner dianggap sebagai media penilaian paling tepat untuk menjalankan pengujian *usability* untuk perangkat lunak. Kuisioner merupakan daftar pertanyaan tertulis yang diberikan kepada subjek yang diteliti untuk mengumpulkan informasi yang dibutuhkan penulis. Menggunakan kuesioner bertujuan untuk mendapatkan penilaian saat proses pengujian.

Kuisioner pada umumnya bermodel tabel. Kuisioner yang terdiri dari baris dan kolom, pada kolom pertama berisi pernyataan yang sesuai dengan kebutuhan penilaian, kemudian kolom selanjutnya berisi tentang skala nilai untuk mengetahui nilai dari setiap pernyataan yang disajikan. Penelitian ini menggunakan kuesioner yang dibentuk dalam skala lima poin dengan model skala Likert untuk memilih jawaban sebagai pengukuran tingkat persetujuan pengguna terhadap pernyataan.

Skala likert digunakan untuk bagaimana mengukur sikap, pendapat, dan persepsi seseorang tentang kejadian. Skala likert merupakan suatu skala yang paling banyak digunakan dalam riset berupa survei (Tjiptono & Diana, 2019).

Saat menanggapi pertanyaan dalam skala likert, responden menentukan tingkat suatu pernyataan dengan memilih salah satu dari pilihan yang tersedia. Pada dasarnya selalu disediakan lima pilihan yang menjadi dasar pilihan skala dengan format sebagai berikut:

1. Skala 1 = Sangat tidak setuju
2. Skala 2 = Tidak setuju
3. Skala 3 = Netral
4. Skala 4 = Setuju
5. Skala 5 = Sangat setuju

Pembobotan jawaban dalam bentuk *score* pada kuisioner. Keuntungan skala likert:

1. Mudah dibuat dan diterapkan.
2. Terdapat kebebasan dalam memasukan pertanyaan- pertanyaan asalkan harus sesuai dengan konteks permasalahan.
3. Jawaban suatu item berupa alternatif, sehingga informasi lebih diperjelas.
4. Reliabilitas pengukuran bisa diperoleh dengan jumlah item.

Untuk mendapatkan hasil interpretasi nilai Y dihitung dengan menggunakan rumus

Y = skor tertinggi skala likert x total responden

Untuk menghitung index digunakan rumus

Index = Total Skor /Y x 100 %

Interval penilaian dari skala likert yaitu

1. Indeks 0% – 19,99% : Sangat Tidak Puas
2. Indeks 20% – 39,99% : Tidak Puas
3. Indeks 40% – 59,99% : Netral
4. Indeks 60% – 79,99% : Puas
5. Indeks 80% – 100% : Sangat Puas