# BAB II
# LANDASAN TEORI

## 2.1 Tinjauan Pustaka

Menurut penelitian yang berjudul Perancangan Sistem Informasi Geografis Objek Wisata Labuhanbatu Berbasis Web, dengan dibangunnya sistem objek wisata Labuhanbatu bertujuan untuk memberikan informasi dan mempermudah wisatawan menemukan jalan menuju lokasi serta meningkatkan Pendapatan Asli Daerah (PAD) dan pendapatan penduduk sekitar. Sistem informasi ini dirancang menggunakan Codeigniter, Xampp, PHP, MySQL, Google Maps, dan Notepad++, sedangkan pengujiannya menggunakan *Blackbox* (Dwi et al., 2021).

Pada penelitian dengan judul Perancangan Sistem Informasi Wisata Kopeng Berbasis Web dengan Metode *Waterfall* menerangkan bahwa dalam proses pembangunan sistem menerapkan beberapa urutan perngerjaan dalam implementasi metode *waterfall*, yaitu : analisis kebutuhan, desain sistem, penulisan kode program dan pengujian program serta penerapan program dan pemeliharaan. Dengan berhasilnya perancangan sistem informasi wisata Kopeng ini berdampak terhadap penyedia dan pelaku wisata yang semakin mudah untuk melakukan promosi wisata (Priyambodo et al., 2022).

Sedangkan penelitian yang berjudul Pembuatan Sistem Penjualan Paket Wisata Kampoeng Batik Laweyan Surakarta membuktikan bahwa dalam meningkatkan penjualan di Kampoeng Batik sistem yang telah dibuat mampu mempengaruhi kepuasan pengguna sebesar 68,9% (Farianti, 2020).

Pada penelitian lain yang berjudul Analisis Tingkat Usabilitas Aplikasi Kamus Istilah Psikologi Menggunakan Metode *System Usability Scale* menyatakan bahwa pengukuran usabilitas akan memberikan gambaran kepada pengembang aplikasi tentang kepuasan pengguna aplikasinya dan memberikan rekomendasi penyempurnaan bagi pengembang aplikasi yang serupa (Ernawati et al., 2023).

## 2.2 Kerangka Pemikiran

Kerangka pemikiran yang dijalankan dalam penelitian ini adalah sebagai berikut :

a. Latar Belakang

Penyebaran informasi seputar pariwisata di Kecamatan Tengaran masih bersifat manual, sehingga potensi wisata kurang terekspos dan tak jarang masyarakat beranggapan bahwa wisata yang berada di wilayah Kecamatan Tengaran merupakan aset milik daerah lain. Oleh karenanya untuk memperkenalkan wisata di wilayah Kecamatan Tengaran dan meningkatkan daya tarik wisatawan perlu direncanakan dan dikelola dengan baik oleh pemerintah daerah setempat agar dapat mengangkat perekonomian maupun kehidupan sosial masyarakat.

b. Rumusan Masalah

Bagaimana tingkat usabilitas Sistem Informasi Potensi Wisata Berbasis Web di Kecamatan Tengaran Kabupaten Semarang ketika diakses pengguna untuk menyebarluaskan informasi pariwisata di Kecamatan Tengaran Kabupaten Semarang.

c. Penguasaan Dasar PHP dan MySQL

Mempelajari dasar-dasar PHP dan MySQL agar lebih menguasai perintah atau program yang nantinya akan digunakan dalam pembuatan sistem informasi.

d. Pengumpulan Data

Pengumpulan data dilakukan dengan cara observasi atau dengan wawancara langsung serta mencari referensi dari buku, jurnal, dan sumber lainnya.

e. Tinjauan Pustaka Penelitian Sebelumnya

Selain mencari sumber referensi dari buku maupun jurnal, penulis juga melakukan studi pustaka terhadap penelitian-penelitian terdahulu yang relevan.

f. Analisis Perancangan Sitem Informasi

Penguasaan analisis dan perancangan sistem akan membantu dalam melakukan analisis dan pembuatan sistem informasi. Analisis sistem dalam penelitian ini meliputi, perancangan aplikasi yaitu *Use Case Diagram, Activity*

*Diagram*, *Sequence Diagram*, *Class Diagram*. Sedangkan perancangan *database* menggunakan MySQL dengan bahasa pemrograman PHP.

g. Implementasi Sistem

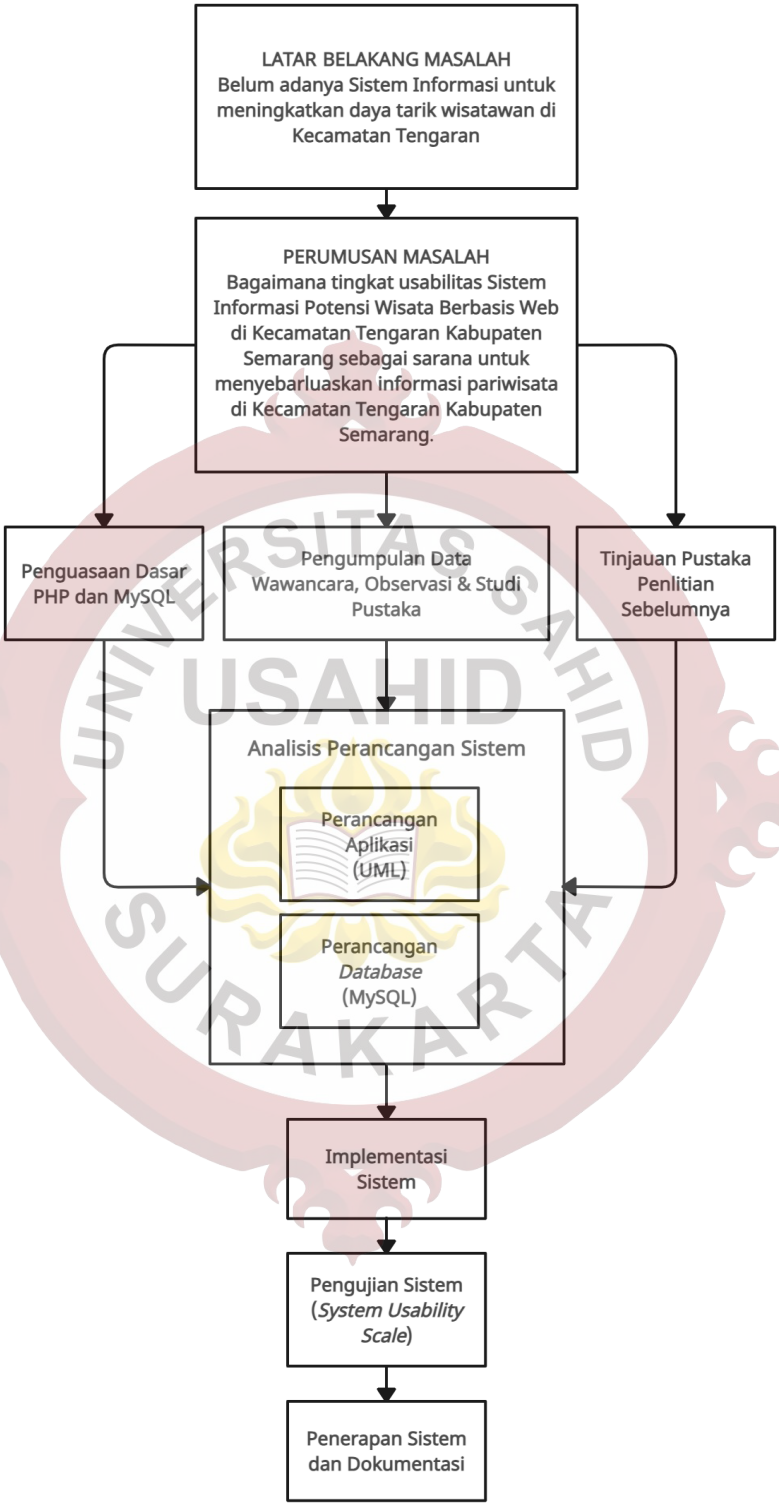
Implementasi Sistem Informasi Potensi Wisata Berbasis Web di Kecamatan Tengaran Kabupaten Semarang melalui prosedur-prosedur yang dilakukan dalam menyelesaikan desain sistem yang telah disetujui. Implementasi *website* ini meliputi kebutuhan perangkat lunak, perangkat keras, tampilan program yang sesuai, *query* yang digunakan, pemrograman, pengujian program dan pengujian *website* yang telah dirancang sesuai dengan kebutuhan dalam pembuatan sistem ini.

h. Pengujian Sistem

Pengujian sistem adalah prosedur yang dilakukan untuk menyelesaikan rancangan sistem yang telah disetujui, menguji sistem, menginstal serta memulai penggunaan sistem baru. Dalam penelitian ini menggunakan penilaian *usabilitas* berdasarkan kemudahan pengguna dalam menggunakan *website*. Data diambil melalui kuesioner yang diberikan kepada responden, kuesioner yang dipakai menggunakan Skala *Likert* dengan metode *System Usability Scale* (SUS).

i. Penerapan dan Dokumentasi

Tahap akhir perancangan sistem ini yaitu penerapan pada Kantor Kecamatan Tengaran Kabupaten Semarang dan dokumentasi guna pelaporan penelitian. Penjabaran kerangka pemikiran sebagaimana Gambar 2.1.




Gambar 2.1 Kerangka Pemikiran Tugas Akhir

## 2.3 Teori Pendukung

### 2.3.1 Sistem Informasi

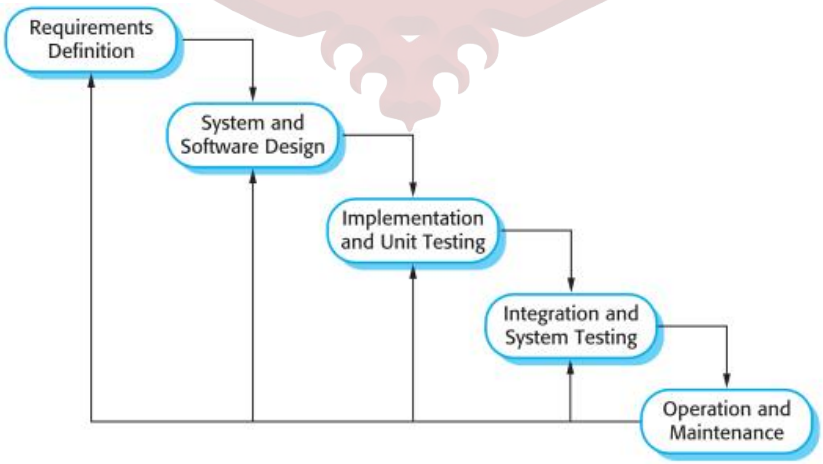
Sistem informasi selalu identik dengan sesuatu yang melibatkan komputer atau sering disebut C*omputer Based Information Systems* atau CBIS. Sistem informasi sendiri secara khusus dapat diartikan suatu sistem yang mencakup sejumlah komponen (manusia, komputer, teknologi informasi dan prosedur kerja), ada sesuatu yang diproses (data menjadi informasi) dan dibuat untuk mencapai suatu tujuan atau sasaran (Kadir, 2018).

### 2.3.2 Wisata

Menurut Undang-undang Republik Indonesia Nomor 10 Tahun 2009 tentang Kepariwisataan menerangkan bahwa wisata adalah suatu kegiatan perjalanan yang dilakukan oleh seseorang atau sekelompok orang dengan mengunjungi tempat tertentu untuk tujuan rekreasi, pengembangan pribadi, atau mempelajari keunikan daya tarik wisata yang dikunjungi. Sehingga sumber daya pariwisata, alam maupun budaya yang potensial menjadi daya tarik wisatawan (UU RI Nomor 10, 2009).

### 2.3.3 *Waterfall*

Waterfall merupakan teori dari sebuah metode pengembangan sistem, dalam penerapannya pendekatan metode ini disusun secara sistematis. Setiap langkah harus diselesaikan satu per satu dan berurutan, oleh karenanya diibaratkan sebagai air terjun atau waterfall (Sommerville, 2011). Gambar 2.2 merupakan langkah-langkah metode *waterfall* yang dikemukakan oleh Ian Sommerville.




Gambar 2.2 Metode Waterfall

j. *Requirements Definition*

Analisis dan definisi kebutuhan sistem, kendala, dan tujuan sebagai perencanaan kebutuhan spesifikasi sistem.

k. *System And Software Design*

Mengidentifikasi sistem dan desain perangkat lunakberupa deskripsi abstraksi yang dibutuhkan.

l. *Implementastion And Unit Testing*

Tahap implementasi dan pengujian sistem. Pengujian sistem berupa verifikasi bahwa setiap unit memenuhi spesifikasi yang telah ditetapkan.

m. *Integration And System Testing*

Tahap integrasi dan pengujian sistem untuk memastikan bahwa kebutuhan perangkat lunak telah terpenuhi.

n. *Operation And Maintance*

Operasi dan pemeliharaan merupakan fase siklus hidup terpanjang. Sistem diinstal dan digunakan secara praktis. Pemeliharaan melibatkan perbaikan kesalahan yang tidak ditemukan pada tahap-tahap sebelumnya.

### 2.3.4 *Website*

*Website* merupakan kumpulan halaman web yang saling berhubungan, pada umumnya berisikan kumpulan informasi dan dapat diakses melalui peramban atau *browser* dan jaringan internet. *Website* disusun berdasarkan halaman utama tunggal, yang dapat diakses dengan memasukkan URL (Pickle, 2023). Kelebihan dari *website* yaitu dapat diakses oleh siapa saja dan kapan saja, sehingga sangat cocok digunakan sebagai media promosi.

### 2.3.5 PHP

PHP atau *Hypertext Preprocessor* merupakan salah satu jenis bahasa pemrograman *web* yang bersifat *open source* yang populer dan sangat cocok untuk pengembangan web. Bersifat cepat, fleksibel, dan pragmatis (PHP, 2024).

PHP adalah *script* yang ditanamkan dalam HTML dan digunakan untuk membuat halaman *website* yang dinamis. *Website* dinamis yaitu web yang dapat menampilkan informasi terbaru dan dapat berinteraksi dengan *user*.

### 2.3.6 HTML

*Hyper Text Markup Language* (HTML) adalah bahasa *markup* standar untuk membuat halaman web, menggambarkan struktur sebuah halaman web yang terdiri dari serangkaian elemen (W3schools, 2024).

### 2.3.7 *Database*

*Database* atau basis data yaitu kumpulan data yang tersimpan secara sistematis, yang nantinya diakses pengguna untuk menampilkan suatu informasi (Indrajani, 2015). Salah satu contoh *database* yaitu MySQL, merupakan sistem manajemen *database* yang bersifat *open-source* yang berfungsi untuk mengelola informasi di *database* pada sisi server dengan bahasa pemrograman SQL.

### 2.3.8 XAMPP

XAMPP merupakan *tool* pembantu pengembangan aplikasi berbasis PHP yang berbasis *open source*. XAMPP mengombinasikan beberapa paket perangkat lunak berbeda ke dalam satu paket. XAMPP juga menyediakan aplikasi *database* MySQL dengan *interface* mudah untuk dioperasikan. Selain tidak berbayar, *tool-tool* yang disediakan cukup lengkap dalam memenuhi kebutuhan perancangan *database* (Apachefriends, 2024).

### 2.3.9 *Bootstrap Framework*

*Bootstrap* adalah salah satu *framework* CSS yang paling populer digunakan untuk membangun *website* yang responsif (Bootstrap, 2024). *Bootstrap* menyediakan berbagai komponen dan fitur yang dapat digunakan untuk memudahkan pembuatan tampilan *website* responsif, seperti *grid system, typography, form, button*, dan lain sebagainya.

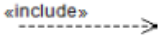
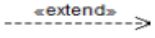
### 2.3.10 *Unified Modeling Language* (UML)

*Unified Modeling Language* (UML) merupakan bahasa pemodelan visual yang mendukung prinsip pengembangan inkremental dan iteratif dari metode pengembangan berorientasi objek. UML membantu untuk membangun model sistem yang jelas dan intuitif, menyatukan komunikasi tim pengembangan, mengurangi kompleksitas pengembangan perangkat lunak, meningkatkan penggunaan kembali dan pemeliharaan sistem, sehingga dapat meningkatkan efisiensi pengembangan perangkat lunak (Kusnadi et al., 2019).

### 2.3.11 *Use Case Diagram*

*Use case diagram* merupakan *tools* yang digunakan untuk mengembangkan sistem dan berfungsi untuk menjelaskan hubungan antar aktor. Sebuah *use case* menjelaskan sebuah fungsi yang disediakan oleh sistem yang menghasilkan *output* yang terlihat untuk seorang aktor. Sedangkan seorang aktor menjelaskan setiap entitas yang berinteraksi dengan sistem. Identifikasi aktor dan *use case* menghasilkan definisi batas sistem (syste*m boundary*), yang membedakan tugas atau kegiatan yang diselesaikan oleh sistem dan tugas atau kegiatan yang diselesaikan oleh lingkungannya. *Actor* berada di luar *system boundary*, sedangkan *use case* berada di dalam *system boundary* (Kusnadi et al., 2019). Simbol *usecase* terdefinisi pada Tabel 2.1.

Tabel 2.1 Simbol Use Case Diagram

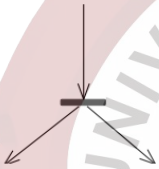
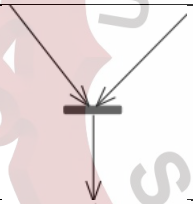
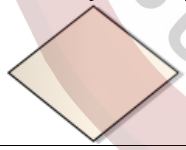
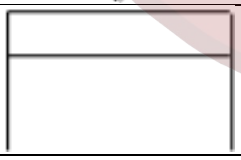
| No | Simbol | Keterangan |
|---|---|---|
| 1. |  | *Actor* atau aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. |
| 2. |  | *Use Case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, yang dinyatakan dengan menggunakan kata kerja. |
| 3. |  | Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengindikasikan data. |
| 4. |  | *Generalization* yaitu hubungan generalisasi danspesialisasi (umum – khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya. |
| 5. |  | *Include* berada di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program. |
| 6. |  | *Extend*, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi. |

### 2.3.12 *Activity Diagram*

*Activity diagram* merupakan salah satu yang paling banyak digunakan di antara semua diagram UML. Alur *activity diagram* dapat digunakan untuk mengontrol urutan tindakan yang dieksekusi atau untuk merepresentasikan

komunikasi data. Biasanya diagram ini digunakan untuk desain tingkat tinggi seperti memodelkan proses bisnis, dan desain tingkat rendah seperti untuk mendeskripsikan algoritma yang akan diimplementasikan (Kusnadi et al., 2019). Simbol *Activity Diagram* sebagaimana Tabel 2.2.

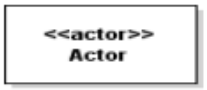
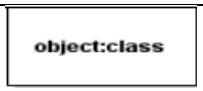
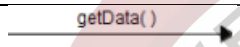
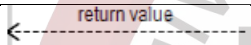
Tabel 2.2 Simbol Activity Diagram

| No | Simbol | Keterangan |
|---|---|---|
| 1. |  | *Start Point*, diletakkan pada pojok kiri atas dan menandakan awal aktivitas. |
| 2. |  | *End Point*, diletakkan pada akhir aktivitas dan menandakan berakhirnya aktivitas. |
| 3. |  | *Activities*, menggambarkan suatu proses/kegiatan bisnis. |
| 4. |  | *Fork*/percabangan, digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel atau untuk menggabung kan dua kegiatan paralel menjadi satu. |
| 5. |  | *Join* (penggabungan) atau *rake*, digunakan untuk menunjukkan adanya dekomposisi. |
| 6. |  | *Decision Points*, menggambar kan pilihan untuk pengambilan keputusan, *true* atau *false*. |
| 7. |  | *Swimlane*, pembagian *activity diagram* untuk menunjukkan siapa melakukan apa. |

### 2.3.13 *Sequence Diagram*

*Sequence diagram* adalah diagram UML paling umum digunakan untuk merepresentasikan bagaimana objek berinteraksi dan bertukar pesan dari waktu ke waktu. *Sequence diagram* menggambarkan bagaimana peristiwa (aktivitas) dalam *usecase* yang dipetakan ke dalam operasi kelas objek dalam *class diagram* (Kusnadi et al., 2019). Keterangan simbol-simbol *Sequence Diagram*, sebagaimana Tabel 2.3.

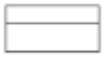
Tabel 2.3 Simbol Sequence Diagram

| No | Simbol | Keterangan |
|---|---|---|
| 1. |  | *Actor* merupakan eksternal sistem berupa orang atau sistem lain yang mendapatkan manfaat dari *usecase* |
| 2. |  | *Object* merupakan sesuatu yang ikut dalam rangkaian kegiatan dengan mengirim atau menerima pesan. |
| 3. |  | *Lifeline* menunjukan waktu hidup suatu objek dalam rangkaian kegiatan. |
| 4. |  | *Execution Occurrence* menandakan ketika suatu objek mengirim atau menerima pesan. |
| 5. |  | *Request Message* merupakan permintaan informasi dari satu objek ke objek lain. |
| 6. |  | *Return Message* merupakan permintaan informasi yang dikirim oleh suatu objek. |

### 2.3.14 *Class Diagram*

*Class Diagram* digunakan untuk memodelkan tampilan struktur sistem yang mencakup kemampuan untuk membawa data dan menjalankan tindakan (*execute actions*), selain itu juga sebagai dasar untuk pengembangan sistem perangkat lunak lebih lanjut (Kusnadi et al., 2019), keterangan simol-simbol *Class Diagram* terdapat pada Tabel 2.4.

Tabel 2.4 Simbol Class Diagram

| No | Simbol | Keterangan |
|---|---|---|
| 1. |  | *Generalization* yaitu hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*). |
| 2. |  | *Nary Association* yaitu upaya untuk menghindari asosiasi dengan lebih dari 2 objek. |
| 3. |  | *Class* yaitu himpunan dari objek-objek yang berbagi atribut serta operasi yang sama. |
| 4. |  | *Collaboration* yaitu deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu *actor*. |
| 5. |  | *Realization* yaitu operasi yang benar-benar dilakukan oleh suatu objek. |

| No | Simbol | Keterangan |
|---|---|---|
| 6. |  | *Dependency* yaitu hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri. |
| 7. |  | *Association* menandakan hubungan antara objek satu dengan objek lainnya. |
| 8. |  | *Directed Association* yaitu relasi antara kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity.* |

2.3.15 *Component Diagram*

*Component diagram* dibuat untuk menunjukkan organisasi dan ketergantungan di antara kumpulan komponen dalam sebuah sistem, yang divisualisasikan pada Tabel 2.5. *Component diagram* fokus pada komponen sistem yang dibutuhkan dan ada di dalam sistem. *Component diagram* juga dapat digunakan untuk memodelkan *source code* program perangkat lunakdan komponen *executable* yang dilepas ke *user* (Kusnadi et al., 2019). Di dalam komponen diagram, terdapat 3 tipe model, yaitu sebagi berikut :

*a. Deployment Component*

Merupakan tipe model yang menjadi basis dari *executable system*. Contohnya *Dynamic Library Link* (DLL), *Active X Control, Java Bean* dan lain sebagainya.

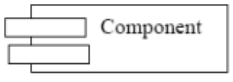
*b. Work Product Component*

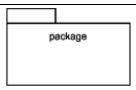
Merupakan salah satu tipe model yang berisi file-file yang dibutuhkan untuk pembuatan *deployment component*. Contohnya file data, file *source code*, dan file penting lainnya.

*c. Execution Component*

Merupakan tipe terakhir dari komponen diagram. Dimana model ini dibuat sebagai hasil dari sistem yang akan dijalankan.

Tabel 2.5 Simbol Component Diagram

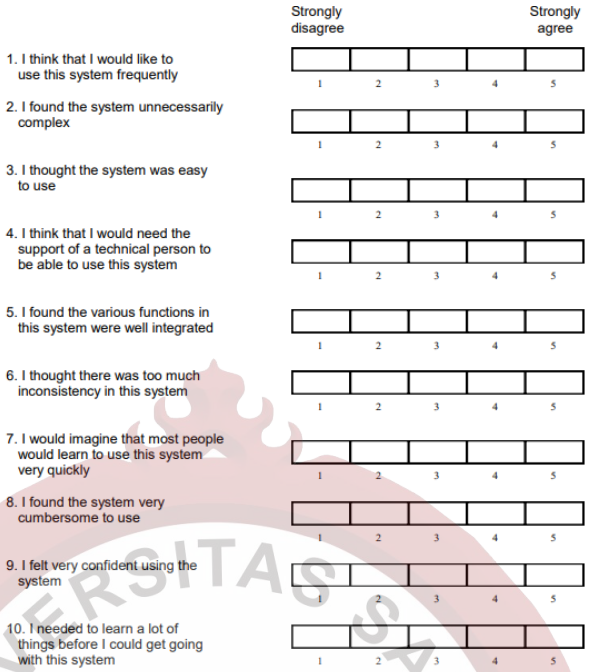
| No | Simbol | Keterangan |
|---|---|---|
| 1. |  | *Component*, yaitu komponen suatu sistem. |

| No | Simbol | Keterangan |
|---|---|---|
| 2. |  | *Depedency*, menandakan kebergantungan antar komponen. Arah panah tersebut mengarah pada komponen yang dipakai. |
| 3. |  | *Interface*, sebagai antarmuka komponen agar tidak mengakses langsung komponen. |
| 4. |  | *Link*, digunakan sebagai relasi antar komponen. |
| 5. |  | *Package* merupakan sebuah bungkusan dari satu atau lebih komponen. |

### 2.3.16 *System Usability Scale* (SUS)

Pengujian *usability* merupakan salah satu standar dalam mengidentifikasi kualitas sistem, pengujian usabilitas diidentifikasi berdasarkan kemudahan pengguna dalam menggunakan *website*, dalam pengujiannya menggunakan kuisioner *System Usability Scale* (SUS).

Menurut Brooke *System Usability Scale* (SUS) menggunakan skala likert (Brooke, 1995). Skala likert didasarkan pada pertanyaan-pertanyaan pilihan yang terbatas pada skala 5. Terdapat beberapa karakteristik dari SUS yang membedakan dari kuesioner lain. Pertama, SUS terdiri dari sepuluh pertanyaan sebagaimana Gambar 2.3., sehingga relatif cepat dan mudah bagi responden untuk menyelesaikan. Kedua, SUS menggunakan teknologi agnostik, yang berarti dapat digunakan secara luas dan mengevaluasi hampir semua jenis *interface*. Ketiga, hasil kuesioner adalah nilai tunggal, mulai dari skor 0 sampai 100, dan relatif mudah dipahami oleh berbagai disiplin, baik individu maupun kelompok.




Gambar 2.3 Item Pertanyaan SUS

Menghitung Skor SUS dapat dilakukan setelah kuesioner SUS disebarkan dan para responden yang telah dipilih telah memberikan penilaian mereka terhadap kesepuluh pertanyaan di dalam kuesioner tersebut. Langkah selanjutnya adalah melakukan proses perhitungan untuk data-data tersebut. Terdapat beberapa aturan dalam melakukan perhitungan skor SUS, diantaranya:

a. Untuk setiap pertanyaan bernomor ganjil, hasil skornya dikurangi angka 1. [Penilaian pengguna – 1 = skor pertanyaan]

   Untuk setiap pertanyaan bernomor genap, maka kita harus mengurangi angka 5 dengan hasil skornya. [5 - Penilaian pengguna = skor pertanyaan]

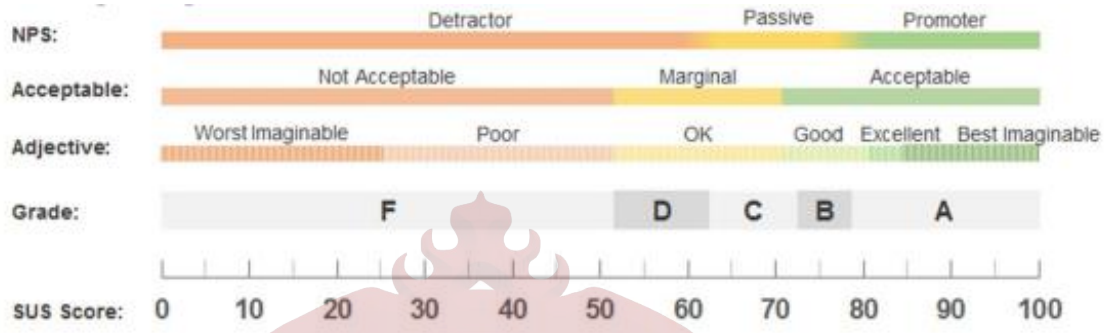
b. Kemudian jumlahkan semua hasil skor dari setiap pertanyaan per responden hasilnya dikalikan dengan angka 2,5.

   [[Skor pertanyaan ke 1] + [Skor pertanyaan ke 2] + … + [Skor pertanyaan ke n] * 2.5 = skor responden]

c. Jumlahkan semua hasil skor setiap responden yang telah melalui langkah 1 hingga 3 diatas, kemudian hitung nilai rata-ratanya.

   [Total skor responden] / jumlah responden = Hasil Skor SUS

Setelah dihitung didapatlah skor rata-rata SUS dari semua responden. Skor tersebut kemudian disesuaikan dengan penilaian SUS. Kategori hasil pengujian dengan skor rata-rata dapat dilihat melalui skala interprestasi pada Gambar 2.3.




Gambar 2.4 Skala Interprestasi SUS (Bangor, 2008)

Tabel 2.6 berikut memberikan gambaran yang lebih jelas mengenai nilai-nilai yang terdapat pada Gambar 2.4 diatas.

Tabel 2.6 Skala Interprestasi SUS (Bangor, 2008)

| *Grade* | SUS | *Percentile range* | *Adjective* | *Acceptable* | NPS |
|---|---|---|---|---|---|
| A+ | 84.1 – 100 | 96 – 100 | *Best Imaginable* | *Acceptable* | *Promoter* |
| A | 80.8 – 84.0 | 90 – 95 | *Excellent* | *Acceptable* | *Promoter* |
| A- | 78.9 – 80.7 | 85 – 89 | | *Acceptable* | *Promoter* |
| B+ | 77.2 – 78.8 | 80 – 84 | | *Acceptable* | *Passive* |
| B | 74.1 – 77.1 | 70 – 79 | *Good* | *Acceptable* | *Passive* |
| B- | 72.6 – 74.0 | 65 – 69 | | *Acceptable* | *Passive* |
| C+ | 71.1 – 72.5 | 60 – 64 | | *Acceptable* | *Passive* |
| C | 65.0 – 71.0 | 41 – 59 | | *Marginal* | *Passive* |
| C- | 62.7 – 64.9 | 35 – 40 | *OK* | *Marginal* | *Passive* |
| D | 51.7 – 62.6 | 15 – 34 | | *Marginal* | *Detractor* |

## 2.4 Spesifikasi Alat yang Digunakan

### 2.4.1 Perangkat Keras

Spesifikasi perangkat keras yang digunakan dalam penelitian ini dapat dilihat pada Tabel 2.7.

Tabel 2.7 Spesifikasi Perangkat Keras yang Digunakan

| Jenis Perangkat Keras | Keterangan |
|---|---|
| Laptop | Lenovo ThinkPad Intel Core i5-3320M with 3 MB L3 cache (2.6 GHz up to 3.6 GHz, 1600 MHz FSB), 4 GB DDR3, Windows 7 64 bit |
| Modem | Modem ZTE F609 |

### 2.4.2 Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan dalam penelitian ini dapat dilihat pada Tabel 2.8.

Tabel 2.8 Spesifikasi Perangkat Lunak yang Digunakan

| No | Jenis Perangkat Lunak | Keterangan |
|---|---|---|
| 1. | Sistem Operasi | *Windows 7 64 bit* |
| 2. | *Web Server* | *XAMPP v3.3.0* |
| 3. | *Web Browser* | *Google Chrome* |
| 4. | *Text Editor* | *Sublime Text* |
| 5. | Pengolah UML | *Astah* |
| 6. | Pengolah Perancangan Tampilan | *Balsamiq Mockup* |

UNIVERSITAS SAHID USAHID SURAKARTA