BAB II
TINJAUAN PUSTAKA DAN LANDASAN TEORI

2.1 Tinjauan Pustaka

Penelitian dengan topik rancang bangun aplikasi menggunakan metode RAD dan mengenai topik reservasi sebuah tempat atau barang tentunya sudah berhasil dilakukan beberapa kali sebelumnya. Tabel 2.1 menunjukkan penelitian terdahulu yang peneliti kumpulkan.

Tabel 2.1 Penelitian Terdahulu

No	Judul Penelitian	Masalah	Metode	Hasil	Perbedaan
1	Perancangan Sistem Informasi Pengelolaan Lapangan Futsal Berbasis Web Dengan Metode *Rapid Application Development* Menggunakan Algoritma *String Matching* Di Maestro Futsal Kemayoran Jakarta, Wahidin dkk, 2021 [5].	Proses reservasi terlalu lama dan terhambat karena proses reservasi harus datang langsung.	*Rapid Application Development*	Web pengelolaan lapangan futsal.	Fitur web yang berbeda dan tidak ada penjelasan mengenai *framework* PHP apa yang digunakan.
2	Penerapan Metode *Waterfall* Dalam Aplikasi Penyewaan Lapangan Futsal Akasia Berbasis Web, Puji Astuti dan Nia Nuraeni, 2019 [6].	Alur reservasi lapangan futsal yang berlaku dirasa kurang efektif karena menguras waktu.	*Waterfall*	Sistem reservasi berbasis web.	Metode penelitian berbeda dan tidak menggunakan *framework*.
3	Sistem Informasi Reservasi Kelas Kesehatan dan Pengelolaan Studio (Studi Kasus Studio *Headspace* Liza Natalia), Harry Agustian dan Yuwan Jumaryadi, 2019 [7].	Proses reservasi tersebut sering ditemui kesalahan saat pencatatan formulir.	*Rapid Application Development*	Aplikasi web dan *mobile*.	Topik reservasi berbeda dan platform aplikasi berbeda.

Penelitian pertama berjudul "Perancangan Sistem Informasi Pengelolaan Lapangan Futsal Berbasis Web Dengan Metode Rapid Application Development Menggunakan Algoritma String Matching Di Maestro Futsal Kemayoran Jakarta" diawali dengan permasalahan terhambatnya proses pemesanan lapangan yang disebabkan karena proses reservasi secara konvensional dan masalah pengelolaan data dengan tulis tangan sehingga pencarian data menjadi tidak efektif. Metode perancangan yang digunakan yaitu *Rapid Application Development* dengan tiga UML berupa *use case*, *activity*, dan *class* diagram. Hasil dari penelitian tersebut berupa web pengelolaan lapangan futsal berbasis web dengan fitur reservasi melalui web dan rekap data secara otomatis oleh sistem [5].

Penelitian kedua berjudul "Penerapan Metode *Waterfall* Dalam Aplikasi Penyewaan Lapangan Futsal Akasia Berbasis Web" memiliki latar belakang masalah berupa tidak efektifnya sistem pemesanan yang sudah berlaku karena menguras waktu dalam proses pencarian jadwal kosong dan pemesanan lapangan futsal. Metode penelitian tersebut adalah metode *Waterfall*. Hasil yang didapatkan pada penelitian tersebut adalah web penyewaan lapangan futsal berbahasa PHP dengan beberapa fitur seperti *login* untuk setiap pemakaiannya, halaman yang menampilkan spesifikasi dan foto lapangan asli, serta formulir untuk memesan lapangan tersebut. Metode pengujiannya dengan metode *Black-box testing* dengan hasil pengujian 3 tidak sesuai dan 2 sesuai [6]. Penelitian yang sejenis juga berhasil diteliti pada tahun 2021 mengenai sistem reservasi lapangan futsal, dengan perbedaan pada metode yang digunakan yaitu *prototype* [8].

Penelitian ketiga berjudul "Sistem Informasi Reservasi Kelas Kesehatan dan Pengelolaan Studio (Studi Kasus Studio *Headspace* Liza Natalia)" memiliki masalah tidak efektifnya sistem reservasi yang sudah berlaku dengan alur memesan melalui pesan singkat atau telepon, lalu admin mencatat ke formulir fisik, kemudian melakukan pembayaran pada saat datang ke lokasi. Proses reservasi tersebut sering ditemui kesalahan saat pencatatan formulir. Metode yang digunakan peneliti tersebut adalah *Rapid*

Application Development. Hasil yang didapatkan dari penelitian tersebut adalah aplikasi berbasis web yang digunakan admin dan aplikasi *mobile* yang digunakan *user*. Pada web admin, terdapat halaman untuk menampilkan dan menambah kelas kesehatan, instruktur, jadwal yang tersedia, data *booking*, dan halaman laporan yang masuk. Dalam aplikasi *mobile user,* terdapat menu utama yang menampilkan kelas, instruktur, dan jadwal, formulir registrasi dengan memilih kelas dan instruktur yang tersedia, dan halaman *my booking*. Metode *Black-box testing* digunakan untuk metode pengujian dengan mengecek setiap menu interaktif dan memiliki hasil yang sesuai [7]. Penelitian mengenai reservasi lain juga berhasil dilakukan pada tahun 2019 mengenai aplikasi reservasi tiket bioskop, aplikasi android untuk reservasi paket wisata pelayaran, aplikasi reservasi restoran berbasis android, dan penelitian pada tahun 2020 tentang reservasi jasa tukang bangunan [3], [9]–[11].

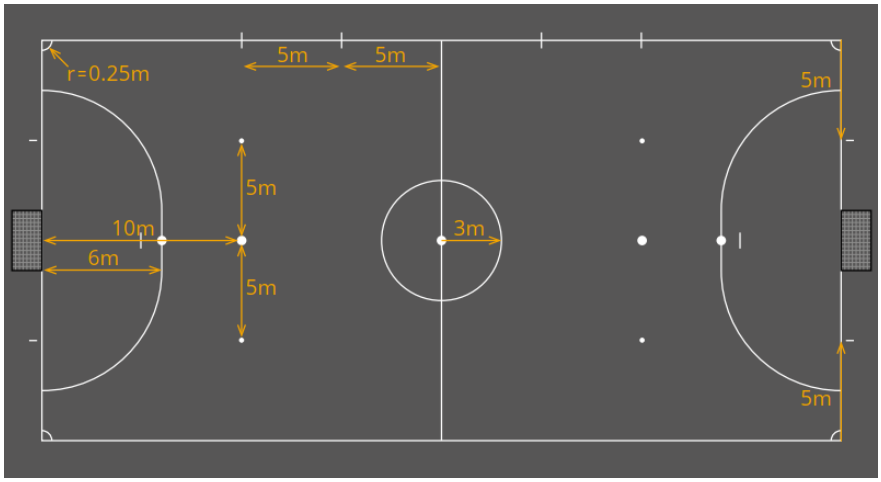
2.2 Landasan Teori

Landasan Teori pada penelitian dibutuhkan untuk memberikan pemahaman tambahan mengenai topik, metode, dan hal-hal yang dibutuhkan dalam penelitian. Berikut beberapa landasan teori yang didapatkan oleh peneliti.

2.2.1 Futsal

Futsal merupakan cabang olahraga dari sepakbola, selain memiliki tujuan yang sama yaitu memasukkan gol ke gawang lawan sebanyak-banyaknya, yang membedakan dari sepakbola adalah lapangan futsal memiliki lapangan yang lebih kecil dibanding lapangan sepak bola. Secara bahasa, istilah futsal, yang merupakan singkatan bahasa Spanyol, *futbol* yang artinya sepakbola dan *sala* yang artinya ruangan. Futsal dapat dimainkan di luar maupun di dalam ruangan [2]. Ukuran lapangan futsal menurut standar FIFA dibagi menjadi 2 jenis, permainan non-internasional memiliki jangkauan panjang 25-42m, lebar 16-25m dan untuk permainan internasional memiliki jangkauan panjang 38-42m dan lebar 20-25m. Gambar 2.1 merupakan gambaran

detail lapangan futsal dengan ukuran selain panjang dan lebar lapangan [12].



Gambar 2.1 Detail Lapangan Futsal

2.2.2 *Website*

World Wide Web *atau WWW* merupakan sebuah layanan ketika pengguna terhubung ke internet, web terdiri dari kumpulan halaman yang digunakan untuk menampilkan data atau informasi berupa gambar, teks, animasi, suara, video atau kombinasi semuanya. Web digunakan sebagai salah satu sarana publikasi informasi yang mudah diakses oleh siapa saja dan di mana saja [13].

Web dapat diklasifikasikan menjadi dua jenis berdasarkan sifatnya, yaitu web statis dan dinamis. Yang dimaksud web statis adalah arah komunikasinya berjalan satu arah antara server ke *client* dan tidak adanya proses komunikasi data dibalik layar. Web dinamis memiliki komunikasi dua arah yang berarti ketika *client* memasukkan sebuah *input*, maka server web tersebut bisa memprosesnya dibalik layar dan akan menghasilkan *output* yang sesuai [14].

2.2.3 Bahasa Penyusun Web

Hypertext Markup Language, kepanjangan dari HTML merupakan bahasa yang dikenali *browser* untuk menampilkan informasi berupa gambar, animasi, teks, dan video. Setiap kali developer merancang sebuah web, perlu dilakukannya penulisan kode-

kode HTML. Kode HTML atau biasa yang disebut dengan *tag* memiliki aturan dan struktur tersendiri. Penulisan *tag* HTML ditulis dengan apitan tanda kurung runcing [15].

CSS, atau kependekan dari *Cascading Style Sheet* merupakan bahasa penyusun web untuk mengendalikan komponen-komponen dari sebuah web, sehingga akan terasa lebih berwarna dan beragam. CSS biasanya digunakan untuk mengubah desain web yang dibuat dengan bahasa HTML, maka dari itu CSS tidak bisa dipisahkan dari HTML [16].

Javascrpt merupakan bahasa pemrograman yang berfungsi untuk memanipulasi interaksi antarmuka pada aplikasi, sehingga menghasilkan web yang dinamis. *Javascript* dibuat oleh Brendan Eich dengan nama awal *Mocha*, lalu diubah menjadi *Livescript*, dan terakhir menjadi nama *JavaScript*. *Javascript* awalnya dibuat hanya untuk *client-side* dalam web, sekarang *Javascript* bisa bekerja layaknya *server-side* dalam sebuah server, selain itu *javascript* sekarang juga bisa bekerja pada program non-web seperti pembaca PDF [17].

PHP atau kependekan dari *Hypertext Preprocessor* adalah suatu bahasa pemrograman bersifat *open source*, dirancang oleh Rasmus Lerdorf pada tahun 1994 yang berfokus pada pengembangan aplikasi berbasis *web*. PHP dirancang untuk web bersifat dinamis yang berarti data dapat berubah tanpa harus mengubah struktur kodenya, sehingga kode php akan diproses oleh *server* lalu ditampilkan ke *client* sesuai permintaan. Web Server menerjemahkan *file* PHP menjadi *file* HTML dan dilanjut dengan penampilan web tersebut oleh *browser* [18].

2.2.4 *Framework*

Framework merupakan sekumpulan instruksi dalam suatu bahasa pemrograman untuk memudahkan developer dalam mengembangkan programnya. Terdapat keuntungan dalam penggunaan *framework* seperti struktur aplikasi menjadi lebih rapi, penghematan waktu, keamanan aplikasi yang lebih, dan mampu menangani beberapa

masalah dengan mudah karena sudah ada *templating*. Selain keuntungannya, terdapat kekurangan dalam penggunaan *framework*, salah satunya adalah terbatasnya saat proses *developing* jika apa yang dibuat tidak terdapat pada *templating framework* [19].

Laravel merupakan *framework* dari bahasa pemrograman PHP sejak tahun 2011 berlisensi MIT. *Laravel* dibuat berdasarkan konsep MVC. *Laravel* merupakan *framework* yang paling banyak mendapat sorotan di *Github* pada tahun 2015. Versi *laravel* terakhir yaitu versi 8,0. *Laravel* dirancang dengan tujuan mengurangi biaya pengembangan, meningkatkan kualitas web, dan pengalaman bekerja dengan *framework* [20].

Bootstrap adalah salah satu *framework* dari CSS yang berfungsi untuk mengembangkan *web* responsif dengan cepat. Pada *framework* ini terdiri dari beberapa *template* untuk membuat *Grid, Typography, Layout,* tabel, dan lainnya. *Bootstrap* juga menyediakan *plugin* berbahasa *javascript* untuk menambah interaksi pada tampilannya seperti *dropdown, modal,* transisi, dan lainnya. Keuntungan dari penggunaan *bootstrap* adalah tampilan web akan otomatis menyesuaikan dari lebar layarnya, sehingga rapi ditampilkan di *device* apa pun. Fitur yang paling terkenal dari *bootstrap* adalah *grid* yang memvisualkan tampilan web seperti sebuah tabel sehingga dapat dengan mudah membagi ukuran dan tata letak komponen tampilannya [21].

2.2.5 Basis Data

Basis data atau *database* merupakan tempat dari banyak informasi dan data yang terhubung satu sama lain sehingga menjadi suatu kesatuan sehingga dapat diolah oleh *software*. Dengan adanya basis data, pemakai bisa membuat, mengolah, dan menyimpan informasi. *Software* untuk mengakses basis data dinamakan sebagai DBMS yang merupakan kependekan dari *Database Management*

System yang fungsi utamanya untuk manipulasi informasi seperti menambah, mengurang, menghapus, dan lainnya [22].

SQL merupakan salah satu bahasa *query* basis data yang bertujuan untuk memanipulasi data. SQL digunakan untuk mengelola basis data seperti membuat, menambah, menghapus, mencari, mengubah data, dan lainnya. Salah satu DBMS berbahasa SQL adalah *MySQL*, sebuah DBMS berlisensi *open source* dan *freeware* [23].

2.2.6 RAD

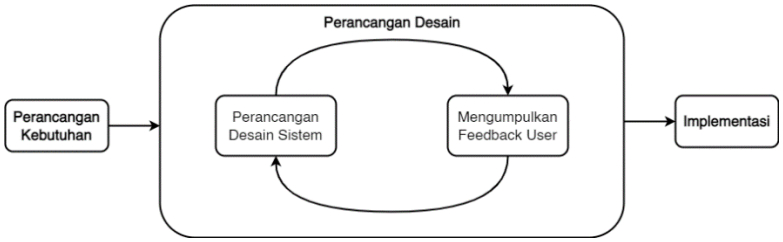
RAD yang merupakan kependekan dari *Rapid Application Development* adalah sebuah SDLC atau alur pembuatan perangkat lunak yang berfokus pada perkembangan secara singkat, sesuai namanya *Rapid*. Proses perencanaan, perancangan, dan implementasi dilaksanakan secara terus-menerus hingga sesuai dengan data yang didapatkan dalam pengumpulan data. Terdapat 3 tahapan dalam SDLC RAD ini, yaitu :

a. Perencanaan Kebutuhan : Developer mengumpulkan data yang didapatkan dari *client* dan mengidentifikasi permasalahan apa saja yang dihadapi oleh *client*, serta membuat tujuan berupa solusi permasalahan yang dihadapi *client*.

b. Perancangan Desain : Developer melakukan analisis terhadap data pada tahap perencanaan lalu membuat rancangan model dan desainnya. Pemodelan sistem dapat dibuat menggunakan diagram sebagai gambaran bagaimana sistem bekerja. Selanjutnya developer membuat desain UI atau *prototype* yang nantinya akan diajukan ke *client* guna mendapati *feedback. Feedback* yang didapat akan dijadikan sebagai model atau desain baru sehingga tahapan ini kembali ke pemodelan sistem. Tahapan ini akan berulang kali hingga *client* tidak memberikan *feedback* atau menganggap sudah layak dilanjut ke tahapan selanjutnya.

c. Implementasi : Developer mengembangkan sistem sesuai dengan hasil desain menggunakan bahasa pemrograman sesuai *platform*

aplikasi yang disepakati. Ketika implementasi selesai, developer melakukan pengujian hingga aplikasi atau sistem yang dibuat layak pakai, yang selanjutnya diteruskan kembali ke *client* [24].



Gambar 2.2 Ilustrasi Metode RAD

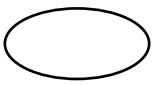
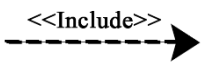
2.2.7 *Unifield Modeling Language (UML)*

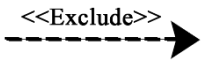
UML adalah sebuah rancangan visual yang digunakan dalam pengembangan sebuah aplikasi. UML digunakan untuk membuat model diagram yang di dalamnya terdapat koneksi antar objek. Menurut Kendall dan Kendall, terdapat beberapa UML yang biasa digunakan dalam pemodelan sistem antara lain [25]:

a. *Use Case* Diagram

Diagram ini digambarkan beberapa jenis aktor dan hal apa saja yang bisa dilakukan di sistem tersebut. *Use Case* juga digunakan untuk menggambarkan batasan setiap aktor dalam memperlakukan sistem. Tabel 2.2 menunjukkan simbol-simbol apa saja yang terdapat pada *Use Case* Diagram [26].

Tabel 2.2 Daftar Simbol *Use Case* Diagram

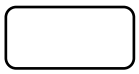
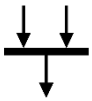
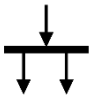
No.	Simbol	Keterangan
1		*Use Case* merupakan gambaran fungsi pada sistem.
2		Aktor merupakan orang atau entitas yang berinteraksi dengan sistem.
3		*Association* sebagai penghubung *use case* dan aktor
4		Relasi yang menunjukkan bahwa *use case* satu bagian dari *use case* lainnya.

No.	Simbol	Keterangan
5		Relasi yang menunjukkan fungsi *use case* tambahan dari *use case* lainnya.

b. *Activity* Diagram

Diagram ini menggambarkan *workflow* dari aktivitas dalam sistem yang sedang dirancang. Aktivitas yang dimaksud adalah dari mana awal aktivitas itu berjalan, *decision* yang terjadi pada sistem, dan bagaimana aktivitas berakhir. Tabel 2.3 menunjukkan simbol-simbol pada *activity* diagram.

Tabel 2.3 Daftar Simbol *Acitvity* Diagram

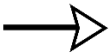
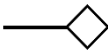
No.	Simbol	Keterangan
1		*Start* digunakan untuk memulai aktivitas.
2		*End* digunakan untuk mengakhiri aktivitas.
3		Aktivitas menunjukkan kegiatan yang dilakukan.
4		*Connector* untuk menghubungkan simbol satu dengan yang lain.
5		*Decision* untuk menggambarkan percabangan sesuai dengan kondisi yang ditentukan.
6		*Join* untuk menggabungkan aktivitas-aktivitas dengan hasil satu aktivitas yang sama.
7		*Fork* untuk membagi satu aktivitas menjadi beberapa aktivitas secara bersamaan.
8		*Swimlane* digunakan untuk mengelompokkan aktivitas berdasarkan aktor.

c. *Class* Diagram

Diagram ini menggambarkan struktur sistem dalam bentuk kelas-kelas pada setiap fungsi yang saling terhubung. Kelas

dibagi menjadi tiga bagian, yang pertama nama kelas, kedua atribut berisi tipe data, nama atribut, dan *visibility*, bagian kelas ketiga adalah metode yang akan digunakan di dalam kelas. Dalam *class* diagram, terdapat relasi yang menghubungkan antar kelas. Tabel 2.4 menunjukkan relasi-relasi yang ada pada *class* diagram [27].

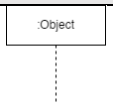
Tabel 2.4 Daftar Relasi *Class* Diagram

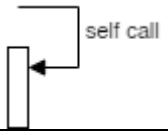
No.	Simbol	Keterangan
1		Asosiasi, adalah relasi umum antar kelas.
2		Generalisasi, adalah relasi dengan makna pewarisan atribut dan metode ke kelas lain. Kelas yang diwariskan tidak bisa berdiri sendiri.
3		Agregasi, adalah relasi salah satu kelas menjadi bagian kelas lain, tetapi kedua kelas bisa berdiri sendiri.

d. *Sequence* Diagram

Diagram ini menampilkan satu skenario dengan bentuk dua dimensi, dimensi vertikal menggambarkan waktu, waktu bertambah semakin ke bawah, sedangkan dimensi horizontal menggambarkan interaksi antar objek. Interaksi antar objek dapat berupa interaksi pada UI yang disebut pesan atau *message.* Tabel 2.5 merupakan daftar simbol yang ada pada *sequence* diagram [28], [29].

Tabel 2.5 Daftar Simbol *Sequence* Diagram

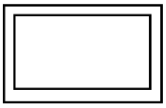
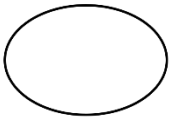
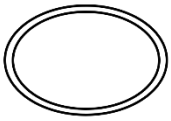
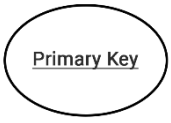
No.	Simbol	Keterangan
1		*Lifeline* berarti entitas objek yang saling berinteraksi.
2		*Activation bar* berfungsi sebagai penanda waktu mulai dan berakhirnya *message*.
3		Aktor menggambarkan pengguna sistem.
4		*Message* menggambarkan interaksi pengiriman pesan ke objek atau *lifeline* selanjutnya.

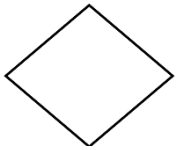
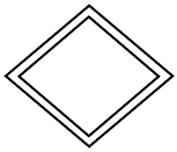
No.	Simbol	Keterangan
5		*Response* menggambarkan interaksi balik pengiriman pesan.
6		*Self Call* menggambarkan interaksi objek dengan dirinya sendiri.

2.2.8 ERD

Entity Relationship Diagram atau disingkat ERD adalah suatu diagram yang berguna untuk menampilkan visualisasi pada basis data kepada pengguna. Tabel 2.6 menunjukkan simbol atau notasi yang terdapat pada ERD.

Tabel 2.6 Simbol Pada ERD

No	Simbol	Keterangan
1		Entitas kuat merupakan simbol yang menggambarkan suatu objek di dunia nyata. Memiliki atribut *key*.
2		Entitas lemah merupakan *entitas* yang bergantung dengan entitas lain. Biasanya ditandai dengan tidak adanya atribut *key*.
3		Atribut merupakan simbol yang menggambarkan informasi berkaitan dengan entitasnya.
4		Atribut bernilai banyak merupakan atribut yang dapat memiliki nilai atau isi banyak. Contohnya hobi.
5		Atribut turunan merupakan atribut yang nilainya dihitung dari atribut lain.
6		*Primary key* merupakan *key* yang menjadi identitas unik sebuah entitas.

No	Simbol	Keterangan
7		Relasi kuat merupakan simbol untuk menggambarkan hubungan antar entitas kuat.
8		Relasi lemah merupakan simbol untuk menggambarkan hubungan antara entitas kuat dan entitas lemah.

ERD memiliki derajat antar entitas yang disebut dengan kardinalitas. Terdapat 3 jenis kardinalitas yaitu:

a. *One to One* : Digambarkan satu entitas A hanya memiliki relasi dengan satu entitas B. Contoh satu provinsi hanya memiliki satu ibukota.
b. *One to Many* : Digambarkan satu entitas A memiliki relasi dengan banyak entitas B. Contoh satu mahasiswa memiliki banyak dosen.
c. *Many to Many* : Digambarkan banyak entitas A memiliki relasi dengan banyak entitas B. Contoh banyak dosen mengajar banyak mata kuliah [30]–[32].

2.2.9 *Black-box Testing*

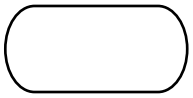
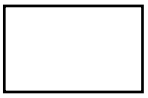
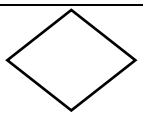
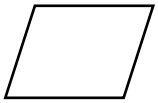
Black-box testing adalah metode pengujian *software* yang berfokus pada pengujian fungsional. Pengujian bertujuan untuk mengidentifikasi fitur-fitur yang dibuat apakah sudah bekerja sesuai harapan developer dengan *input output* dari perangkat lunak tersebut sesuai dengan spesifikasi yang dirancang. Pengujian dilaksanakan dengan membuat sebuah uji kasus dengan perbandingan hasil antara bekerja dengan baik dan bekerja tidak baik. Penguji tidak diwajibkan untuk memahami bahasa pemrograman untuk melakukan *black-box testing* [33].

2.2.10 *Flowchart*

Flowchart atau diagram alir merupakan suatu gambaran dari alur atau langkah-langkah secara berurutan dalam sebuah kegiatan. Dalam

pemrograman, *flowchart* membantu seorang programmer atau analyst untuk menyelesaikan masalah karena proses tersebut dibagi menjadi bagian-bagian kecil. *Flowchart* memiliki beberapa simbol, Tabel 2.7 menunjukkan simbol yang biasanya ada pada *flowchart* [34].

Tabel 2.7 Simbol pada *Flowchart*

No	Simbol	Keterangan
1		*Terminator*, digunakan untuk memulai dan mengakhiri sebuah proses.
2		*Processing*, menunjukkan sebuah proses dalam komputer.
3		*Decision*, menunjukkan percabangan sesuai dengan kondisi yang ada.
4		*Input-Output*, menunjukkan tahapan ketika *user* memasukkan sesuatu dan atau sistem menampilkan sesuatu.

2.2.11 Populasi dan Sampel

Populasi merupakan suatu bagian yang menjadi subjek atau objek sehingga menjadi salah satu elemen pada penelitian yang dilakukan. Populasi tidak selalu orang, akan tetapi semua hal yang terdapat di alam. Populasi juga bisa hanya dengan satu orang, karena karakteristik yang dimilikinya sesuai dengan apa yang diteliti.

Sampel merupakan bagian dari populasi yang nantinya diproses menggunakan metode tertentu dan dianggap menjadi suatu representasi dari populasi pada penelitian. Dalam sebuah penelitian dengan jumlah populasi yang besar, tidak mungkin mempelajari semua anggota populasi dikarenakan keterbatasan dana, waktu, dan tenaga, sehingga sering sekali menerapkan metode *sampling* untuk mendapatkan sampel yang dianggap representatif dari populasi besar tersebut [35].

2.2.12 *Metode Sampling*

Purposive Sampling adalah salah satu metode pengambilan sampel yang termasuk dalam kategori teknik *Nonprobability sampling*. *Nonprobability sampling* sendiri berarti setiap anggota populasi yang dipilih tidak memiliki peluang yang sama untuk dipilih menjadi sampel. *Purposive sampling* mendapatkan sebuah sampel dengan menggunakan pertimbangan-pertimbangan yang sesuai dengan tujuan dan masalah penelitian sehingga sampel dianggap layak untuk dipilih.

Sampling Jenuh adalah metode lain dalam pengambilan sampel jika populasinya kecil atau kurang dari 30, sehingga populasi digunakan sebagai sampel. Istilah lain *sampling* jenuh adalah sensus [35].

2.2.13 Metrik Kinerja

Metrik kinerja digunakan untuk menguji sejauh mana sistem atau produk yang digunakan dalam menyelesaikan tugas tertentu. Metrik kinerja mengandalkan perilaku pengguna dan skenario tugas dalam pengukurannya. Berdasarkan ISO/IEC 9126, terdapat beberapa aspek yang dapat dinilai berkaitan dengan pengalaman pengguna dan metrik kinerja:

a. Efektivitas

Efektivitas merupakan aspek kemampuan menyelesaikan sebuah tugas. Efektivitas dapat diukur dengan menghitung berapa banyak tugas yang berhasil dicapai. Dalam menguji sebuah efektivitas produk, jumlah penguji yang optimum adalah 11-15 partisipan. Penilaian menggunakan nilai biner 1 jika berhasil, dan 0 jika gagal. Efektivitas dapat dihitung menggunakan Persamaan 2.1 dengan $\bar{E}$ adalah *completion rate,* N adalah total tugas, R adalah jumlah partisipan, dan n_{ij} adalah hasil dari tugas, bernilai 1 atau 0.

$$\bar{E} = \frac{\sum_{j=1}^{R}\sum_{i=1}^{N} n_{ij}}{RN} \times 100\% \qquad (2.1)$$

Selanjutnya nilai *completion rate* diinterpretasikan tingkat keefektifan sesuai dengan Tabel 2.8.

Tabel 2.8 Tingkat Efektivitas

No.	Nilai $\bar{E}$	Tingkat Efektivitas
1	≤50%	Sangat Buruk
2	50-75%	Buruk
3	75-90%	Normal
4	90-100%	Baik

b. Efisiensi

Efisiensi merupakan aspek penentuan jumlah upaya yang dilakukan dalam penyelesaian tugas. Efisiensi dapat diukur dalam perhitungan waktu yang dibutuhkan untuk menyelesaikan suatu tugas. Persamaan 2.2 digunakan untuk menghitung efisiensi suatu produk berdasarkan waktu penyelesaian tugas Dengan $\overline{P_t}$ adalah *time-based efficiency*, N adalah total tugas, R adalah jumlah partisipan, n_{ij} adalah hasil dari tugas, bernilai 1 atau 0, dan t_{ij} adalah waktu yang dibutuhkan dalam menyelesaikan tugas..

$$\overline{P_t} = \frac{\sum_{j=1}^{R}\sum_{i=1}^{N}\frac{n_{ij}}{t_{ij}}}{NR} \quad (2.2)$$

c. Kepuasan

Kepuasan biasanya didapatkan atau diukur dengan pengumpulan kuesioner. Kepuasan memiliki perbedaan dengan efektivitas dan efisiensi yaitu dalam kepuasan lebih mengarah pada pendapat subjektif setiap responden. Kepuasan dapat diukur dengan skala kepuasan. Salah satu skala yang digunakan yaitu Skala Likert. Skala Likert adalah skala untuk mengukur pendapat seseorang terhadap survei opini. Pada skala ini, responden menentukan tingkat persetujuan pada beberapa pernyataan yang dibuat oleh peneliti. Sebagai contoh terdapat 5 tingkat persetujuan dengan nilai 1 sangat tidak setuju, 2 tidak setuju, 3 kurang setuju,

4 setuju, dan 5 sangat setuju. Selanjutnya respons tersebut dihitung total skornya menggunakan Persamaan 2.3 dengan w_i adalah bobot skala, n_i adalah jumlah partisipan yang menjawab skala tersebut, dan x adalah banyaknya skala.

$$total\ skor = \sum_{i=1}^{x} w_i \times n_i \quad (2.3)$$

Selanjutnya menghitung persentase indeks menggunakan Persamaan 2.4 [36].

$$Persentase\ indeks = \frac{total\ skor}{skor\ maksimal} \times 100\% \quad (2.4)$$