RANCANGAN *PROTOTYPE* BANGUN SISTEM INFORMASI PERPUSTAKAAN BERBASIS WEB PADA MTsN 2 MAJENE

Oleh
Muh. Aminuddin
H131 15 509

PROGRAM STUDI SISTEM IN*FORM*ASI
DEPARTEMEN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS HASANUDDIN
2022

RANCANGAN *PROTOTYPE* BANGUN SISTEM INFORMASI PERPUSTAKAAN BERBASIS WEB PADA MTsN 2 MAJENE

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Sains pada Program Studi Sistem Informasi Departemen Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin

Muhammad Aminuddin

H13115509

PROGRAM STUDI SISTEM INFORMASI DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

2022

PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini:

Nama	: Muhammad Aminuddin
NIM	: H13115509
Program Studi	: Sistem Informasi
Jenjang	: S1

Menyatakan dengan ini bahwa karya tulisan saya berjudul

Rancangan Prototype Bangun SIstem Informasi Perpustakaan Berbasis Web Pada MTsN 2 Majene

adalah karya tulisan saya sendiri dan bukan merupakan hasil plagiat dan belum pernah dipublikasikan dalam bentuk apapun.

Apabila dikemudian hari terbukti atau dapat dibuktikan bahwa sebagian atau keseluruhan skripsi ini hasil karya orang lain, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Makassar, 24 Agustus 2022

Menyatakan,

Muhammad Aminuddin

NIM: H13115509

RANCANG PROTOTYPE BANGUN SISTEM INFORMASI PERPUSTAKAAN BERBASIS WEB PADA MTSN 2 MAJENE

Disusun dan diajukan oleh

Muhammad Aminuddin

H13115509

Telah diperhatikan di hadapan Panitia Ujian yang dibentuk dalam rangka Penyelesaian Studi Program Sarjana Program Studi Sistem Informasi Fakultas Matematika dan Ilmu Pengetahuan Alam pada Agustus 2022 dan dinyatakan telah memenuhi syarat kelulusan

Menyetujui,

Pembimbing Utama,

Dr. Muhammad Hasbi, M.Sc.
NIP. 196307201989031003

Pembimbing Pertama,

Dr. Hendra, S.Si., M.Kom.
NIP. 197601022002121001

Ketua Program Studi,

Dr. Muhammad Hasbi, M.Sc.
NIP. 196307201989031003

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh:

Nama	:	Muhammad Aminuddin
NIM	:	H13115509
Program Studi	:	Sistem Informasi
Judul Skripsi	:	Rancang Prototype Bangun Sistem Informasi Perpustakaan Berbasis Web Pada MTsN 2 Majene

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Komputer pada Program Studi Sistem Informasi Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin.

		Tanda Tangan
Ketua	: Dr. Muhammad Hasbi, M.Sc.	(.....)
Sekretaris	: Dr. Hendra, S.Si., M.Kom.	(.....)
Anggota	: Naimah Aris, S.Si., M.Math.	(.....)
Anggota	: Edy Saputra Rusdi, S.Si., M.Si	(.....)

Ditetapkan di : Makassar

Tanggal : 24 Agustus 2022

KATA PENGANTAR

Puji syukur penulis panjatkan kehadirat Allah SWT. karena atas berkat dan rahmat-Nya penulis dapat menyelesaikan skripsi ini. Penulisan skripsi ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Sarjana Strata 1 yang berjudul **"Rancang Prototype Bangun Sistem Informasi Perpustakaan Berbasis Web Pada MTsN 2 Majene"**. Saya menyadari bahwa, tanpa bantuan dan bimbingan dari berbagai pihak, dari masa perkuliahan sampai pada penyusunan skripsi ini. Oleh karena itu, pada kesempatan ini dengan segala kerendahan hati penulis menyampaikan terima kasih yang setulus-tulusnya kepada:

1. Keluarga, ayahanda tercinta **Drs. Syamsuddin** dan ibunda tersayang **Dra. Hj. Radhiah, M.Pd.I**. yang telah memberikan dukungan baik moril maupun materiil serta doa yang tiada henti-hentinya kepada penulis.
2. Rektor Universitas Hasanuddin beserta jajarannya, Bapak Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam beserta jajarannya, dan seluruh pihak birokrasi atas pengetahuan yang diberikan, baik dalam bidang akademik maupun bidang kemahasiswaan.
3. Bapak **Bapak Dr. Muhammad Hasbi, M.Sc**. selaku pembimbing utama sekaligus Ketua Program Studi Sistem Informasi dan Bapak **Dr. Hendra, S.Si., M.Kom**. selaku pembimbing pertama untuk segala ilmu dan kesabaran dalam membimbing dan mengarahkan penulis, serta bersedia meluangkan waktunya untuk mendampingi penulis sehingga skripsi ini dapat terselesaikan. Serta kepada Ibu **Naimah Aris, S.Si., M.Math.** dan Bapak **Edy Saputra Rusdi, S.Si., M.Si.** atas kesediaannya menjadi anggota tim penguji yang telah bersedia meluangkan waktunya untuk memberikan saran dan arahan kepada penulis dalam penyusunan skripsi ini.
4. Dosen Departemen Matematika, dan terkhusus kepada ibu dan bapak dosen Program Studi Sistem Informasi Fakultas MIPA Universitas Hasanuddin untuk semua ilmu yang telah diberikan kepada penulis selama menempuh pendidikan di jenjang strata.
5. Pegawai dan staf Science Building dan Departemen Matematika Fakultas MIPA Universitas Hasanuddin yang senantiasa membantu pengurusan dokumen.
6. Teman-teman seperjuangan dari Program Studi Sistem Informasi dan Tehnik Informatika 2015 khususnya kepada **Rosihan Ardiansyah S.T** dan **Ilham, S.Kom.** yang senantiasa mendukung, memberikan semangat, pelajaran-

pelajaran dan motivasi hidup, **Ricky Hendrawan Tuyuwale** dari Program Studi Geofisika 2015**.** yang senantiasa meluangkan waktunya untuk belajar bersama, **Ambo Ecce S.E** dari Program Studi Ekonomi 2015**.** yang senantiasa mmberikan kabar informasi dan sudut pandang berbeda dari fakultas lain.

7. Kakak-kakak dan adik-adik Program Studi Sistem Informasi 2014, 2016, 2017, 2018, 2019, 2020,2021, dan 2022.
8. Komunitas Stack Overflow dan Github yang telah membantu penulis dan programmer lain di belahan dunia lainnya untuk menemukan jalan keluar dari setiap permasalahan dalam menyusun kode program.
9. Semua pihak yang telah membantu penulis dan tak sempat penulis tuliskan satu persatu.

Akhir kata, saya berharap Tuhan Yang Maha Esa berkenan membalas segala kebaikan semua pihak yang telah membantu. Semoga skripsi ini membawa manfaat bagi pengembangan ilmu.

Makassar, 1 Agustus 2022

Penulis

PERNYATAAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai civitas akademik Universitas Hasanuddin, saya yang bertanda tangan di bawah ini:

Nama	: Muh. Aminuddin
NIM	: H13115509
Program Studi	: Sistem Informasi
Departemen	: Matematika
Fakultas	: Matematika dan Ilmu Pengetahuan Alam
Jenis Karya	: Skripsi

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Hasanuddin **Hak Bebas Royalti Non Eksklusif (Non-exclusive Royalty Free Right)** atas karya ilmiah saya yang berjudul:

"Rancang Prototype Bangun Sistem Informasi Perpustakaan Berbasis Web pada MTsN 2 Majene"

beserta perangkat yang ada (jika diperlukan). Terkait dengan hal di atas, maka pihak universitas berhak menyimpan, mengalih-media/format-kan, mengelola dalam bentuk pangkalan data (database), merawat, dan mempublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.
Dibuat di Makassar pada tanggal 24 Agustus 2022

Yang menyatakan

Muh. Aminuddin

ABSTRAK

Di era industri 4.0 saat ini, internet mengambil peranan penting dalam hampir segala bidang kehidupan. Hal ini didukung oleh mudahnya mengakses internet. Tidak terkecuali dalam kegiatan prasarana yang dilakukan di sekolah. Civitas akademika dapat mengakses bahkan mulai mengontrol jalannya proses kegiatan yang dilaksanakan di sekolah tanpa harus ke sana. Hal ini berlaku pula dalam kegiatan transaksi simpan-pinjam buku dalam perpustakaan sekolah. Perpustakaan membutuhkan sebuah sistem informasi. Sistem informasi adalah suatu sistem di dalam suatu organisasi yang membutuhkan kebutuhan pengolahan data transaksi harian, mendukung operasi bersifat manajerial, dan kegiatan strategi dari sebuah organisasi serta menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan. Tujuan penelitian ini ialah menghasilkan sistem informasi perpustakaan yang memberikan solusi dan sarana alternatif untuk membantu civitas akademika Madrasah Tsanawiyah Negeri 2 Majene yang menjadi perwakilan dalam penelitian ini. Dihasilkan Sitem informasi perpustakaan MTsN 2 Majene dengan nilai useabily yang meliputi *usefullness*, *ease of use*, *ease of learning*, dan *satisfaction* dengan skor total 68,2 % yang termasuk kategori cukup baik.

Kata Kunci: Perpustakaan Sekolah, Sistem informasi, Transaksi, Pelaporan. *usefullness*, *ease of use*, *ease of learning*, *satisfaction*.

ABSTRACT

In the current industrial era 4.0, the internet plays an important role in almost all areas of life. This is supported by easy access to the internet. No exception in the infrastructure activities carried out in schools. The academic community can access and even begin to control the process of activities carried out at school without having to go there. This also applies to the activities of saving and borrowing books in the school library. Libraries need an information system. An information system for organization that requires daily transaction data processing needs, supports managerial operations, and strategic activities of an organization and provides certain outside parties with the necessary reports. The purpose of this research is to produce a library information system that provides alternative solutions and means to assist the academic community of Madrasah Tsanawiyah Negeri 2 Majene who are representatives in this study. The resulting MTsN 2 Majene library information system with a useability value which includes usefulness, ease of use, ease of learning, and satisfaction with a total score of 68.2% which is categorized as good enough.

Kata Kunci: school library, information system, transaction, reports. *usefullness*, *ease of use*, *ease of learning*, *satisfaction*.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Di era industri 4.0 saat ini, Internet mengambil peranan penting dalam hampir segala bidang kehidupan. Hal ini didukung oleh mudahnya mengakses internet tersebut kapan dan dimana saja. Tidak terkecuali dalam kegiatan prasarana yang dilakukan disekolah. walaupun tidak sedang berada pada lokasi sekolah tersebut. Civitas akademika dapat mengakses bahkan mulai mengontrol jalannya proses kegiatan yang dilaksanakan disekolah tanpa harus ke sana. Seperti melakukan kegiatan mengajar-mengajar bahkan yang bersifat administratif seperti pengesahan berkas dan lain lainnya.

Hal ini berlaku pula dalam kegiatan transaksi simpan-pinjam buku dalam perpustakaan sekolah. Sebagaimana hakikat perpustakaan sekolah ialah tempat kumpulan buku-buku atau tempat buku-buku dihimpun dan diorganisasikan sebagai media belajar siswa (Darmono, 2001). Dimana perpustakaan pula mengambil peranan besar disekolah semenjak pemerintah mulai menerapkan program wajib belajar 9 tahun, pihak sekolah diberikan amanah agar menggunakan dana Operasionalnya untuk menyediakan buku cetak pada siswa dan gurunya yang secara tak langsung akan dialihkan tanggung jawab terorganisirnya kepada biro perpustakaan ini.

Dilihat dari perannya yang begitu penting dalam sebuah sekolah. Sudah sewajarnya perpustakaan ini perlu perhatian khusus, dan bahkan memiliki poin khusus menyangkut penilaian akreditasi sebuah sekolah. Adanya pegawai dan pejabat-pejabat tertentu yang khusus untuk mengurus kegiatan dalam siklus peredaran buku tersebut secara baik dan diharapkannya merta dan sampai ke setiap civitas akademika yang bersangkutan. Agar mampu mendapat poin penuh dalam aspek perpustakaan ini, kinerja dalam perpustakaan diharapkan jelas dan terstruktur.

Dikarenakan kebutuhan sifat terstruktur ini, sehingga. Perpustakaan membutuhkan sebuah sistem in*form*asi. Sistem in*form*asi adalah suatu sistem di dalam suatu organisasi yang membutuhkan kebutuhan pengolahan data transaksi harian, mendukung operasi bersifat manajerial, dan kegiatan strategi dari sebuah organisasi serta menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan (Yakub, 2012:17).

Perpustakaan MTsN 2 Majene ini dalam melayani para peminjam dan pembaca buku mengalami kendala dalam ranah administrasi di perpustakaan. Hal tersebut meliputi pendataan buku serta transaksi yang sering terjadi kekeliruan. Ditambah lagi. dikarenakan masih bersifat manual, maka siswa harus mengecek terlebih dahulu apakah buku tersebut masih tersedia atau sedang dipinjam.

1.2 Rumusan masalah

Adapun rumusan masalah dalam penelitian ini ialah bagaimana membangun sistem in*form*asi peminjaman buku di Perpustakaan berbasis *Website* yang dapat membantu proses pengelolaan data administrasi serta penyampaian in*form*asi perpustakaan di MTsN 2 Majene ?

1.3 Batasan Masalah

Berikut ini merupakan beberapa batasan dalam penelitian ini.

1. Objek penelitian ini adalah Perpustakaan Madrasah Tsanawiyah Negeri 2 Majene.
2. Aplikasi yang dibangun mencakup ruang lingkup sistem peminjaman, pengembalian, data anggota, data status buku, data inventaris perpustakaan denda keterlambatan, dan in*form*asi koleksi buku.
3. Pengelolaan data administrasi dikelola oleh administratif namun dipantau penuh oleh supervisor/penanggung jawab perpustakaan Madrasah Tsanawiyah Negeri 2 Majene.

1.4 Tujuan Penelitian

Adapun tujuan penelitian ini ialah menghasilkan sistem in*form*asi perpustakaan yang memberikan solusi dan sarana alternatif untuk membantu civitas akademika Madrasah Tsanawiyah Negeri 2 Majene.

1.5 Manfaat penelitian

Berikut beberapa manfaat dari penelitian ini

1. Untuk mempermudah pengaksesan data dan in*form*asi perpustakaan
2. Untuk mempermudah pegawai perpustakaan untuk melakukan pendaftaran, peminjaman, dan pengembalian buku.
3. Penelitian ini diharapkan dapat menjadi rujukan untuk membangun sistem in*form*asi dalam ranah lainnya.

1.6 Organisasi Skripsi

Sistematika penulisan skripsi ini adalah sebagai berikut :

BAB I``: Pendahuluan

Bab ini membahas mengenai latar belakang masalah, rumusan masalah, batasan masalah, tujuan dan manfaat penulisan, serta organisasi skripsi.

BAB II : Tinjauan Pustaka

Bab ini membahas mengenai landasan teori, konsep dasar yang mendasari pokok permasalahan dalam tulisan ini. Serta penelitian terkait.

BAB III : Metodologi Penelitian

Bab ini berisi tahapan penelitian, waktu dan tempat penelitian, sumber data, rancangan sistem, Diagram Alur Aplikasi, dan

instrumen penelitian.

BAB IV : Hasil dan Pembahasan

Bab ini menguraikan tentang perancangan solusi serta implementasi dari masalah- masalah yang telah dianalisis. Pada bagian ini juga akan ditentukan bagaimana sistem dirancang, dibangun, diuji, dan disesuaikan dengan hasil penelitian.

BAB V : Penutup

Bab ini berisi kesimpulan dan saran yang merupakan jawaban yang melatar belakangi masalah pada Bab 1, dan saran untuk perbaikan menindak lanjuti hasil penelitian yang nantinya akan berguna bagi pengembangan sistem ini ke depan.

BAB II
TINJAUAN PUSTAKA

2.1 Definisi Perpustakaan

Berikut ini merupakan pengertian perpustakaan menurut ahli perpustakaan dan sumber lain, diantaranya:

1. Menurut IFLA (*International of Library Associationsand Institutions*) "Perpustakaan merupakan kumpulan bahan tercetak dan non tercetak dan atau sumber in*form*asi dalam komputer yang tersusun secara sistematis untuk kepentingan pemakai."
2. Menurut Sutarno NS, MSi. "perpustakaan adalah suatu ruangan, bagian dari gedung/bangunan, atau gedung itu sendiri, yang berisi buku-buku koleksi, yang disusun dan diatur sedemikian rupa sehingga mudah dicari dan dipergunakan apabila sewaktu-waktu diperlukan untuk pembaca."
3. Menurut C. Larasati Milburga, dkk "perpustakaan adalah suatu unit kerja yang berupa tempat menyimpan koleksi bahan pustaka yang diatur secara sistematis dengan cara tertentu untuk digunakan secara berkesinambungan oleh pemakainya sebagai sumber in*form*asi."
4. Menurut Kamus Besar Bahasa Indonesia (KBBI) Perpustakaan berasal dari kata dasar "pustaka" yang berarti pustaka atau buku. "Perpustakaan" artinya kumpulan buku (bacaan dsb); bibliotek
5. Dalam UU No.43 tahun 2007 tentang perpustakaan disebutkan bahwa: Perpustakaan adalah institusi pengelola koleksi karya tulis, karya cetak, dan/atau karya rekam secara profesional dengan sistem yang baku guna memenuhi kebutuhan pendidikan, penelitian, pelestarian, in*form*asi, dan rekreasi para pemustaka.
6. Perpustakaan sebagai akumulasi bahan pustaka dalam arti luas serta forum yang merupakan titik temu antara pemakai in*form*asi dengan pustakawan

sebagai sumber yang menyediakan jasa temu balik yang efisien dan efektif (Sulistyo dan Basuki, 2005).

Secara garis besar, ada kesamaan dalam lima pengertian perpustakaan tersebut, yaitu kumpulan buku yang diatur secara sistematis. Oleh sebab itu, mengatur buku-buku dengan baik dan sistematis merupakan hal paling dasar dalam penataan ruang utama perpustakaan. Sehingga dapat disimpulkan Perpustakaan adalah sebuah gedung atau akomodasi fisik tempat menyimpan buku dan media non-buku, digital maupun analog.

2.2 Konsep Dasar Sistem In*form*asi

Menurut Jogiyanto (2005) Sistem In*form*asi adalah : "Suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan". Adapun juga menurut Ladjamuddin (2005) Sistem In*form*asi adalah: "suatu sistem yang dibuat oleh manusia yang terdiri dari komponen-komponen dalam organisasi untuk mencapai suatu tujuan yaitu menyajikan in*form*asi".

Berdasarkan uraian di atas, dapat disimpulkan bahwa sistem in*form*asi merupakan suatu sistem yang mengatur alur pengolahan transaksi semua anggota dan komponen-komponen pendukungnya dalam menyampaikan sebuah in*form*asi dengan bentuk laporan-laporan tertentu.

2.3 Konsep Dasar *Website*

Berikut ini merupakan pengertian *Website* menurut ahli dan tenaga pengajar lainnya, diantaranya:

Website merupakan kumpulan halaman-halaman yang berhubungan dengan *file-file* lain yang saling terkait. Dalam sebuah *website* terdapat satu halaman yang dikenal dengan sebutan *home-page*. *Homepage* adalah sebuah halaman yang

pertama kali dilihat ketika seseorang mengunjungi sebuah *website*. (Jhonsen, 2004). Adapun World Wide Web (www) secara luas lebih dikenal dengan istilah web (*website*). Web adalah sistem pengakses in*form*asi dalam internet (Kadir, 2014).

Web disusun dari halaman – halaman yang menggunakan teknologi web dan saling berkaitan satu sama lain. Sedangkan pengertian lain menyebutkan bahwa website adalah rangkaian atau sejumlah halaman web di internet yang memiliki topik saling berkaitan untuk mempresentasikan suatu in*form*asi (Ginanjar, 2014). Web dan internet merupakan dua hal yang berbeda. Internet lebih merupakan perangkat keras dan web merupakan perangkat lunak. Protokol yang digunakan internet dan web berbeda, internet menggunakan TCP/IP sebagai *protocol* sedangkan web menggunakan HTTP (*Hyper Text Transfer Protocol*) (Suharto, 2012).

Website memiliki beberapa jenis, jenis-jenis tersebut dikelompokkan berdasarkan sifat dan Bahasa pemrograman yang digunakan. Halaman *web* dapat digolongkan menjadidua yaitu :

a. *Website* Statis

   Website statis merupakan *web*site yang berisikan data dan in*form*asiyang tidak berubah – ubah. Dokumen *web* yang di kirm kepada *client* akan sama isinya dengan *web server*. Contohnya adalah halaman utama *Google* karena tidak adanya perubahan data atau in*form*asi.

b. *Website* Dinamis

   Website dinamis merupakan *website* yang memiliki data dan in*form*asi yang berbeda – beda tergantung *input* yang disampaikan oleh *client*. Contohnya adalah pada *Google* ketika sedang melakukan pencarian.

2.4 HTML

HTML (*Hyper Text Markup Language*) adalah suatu *form*at data yang digunakan untuk membuat dokumen *hypertext* yang dapat dieksekusi dari satu plat*form* komputer ke plat*form* komputer lainya tanpa perlu melakukan suatu

perubahan apapun dengan suatu alat tertentu. Generasi baru dari HTML adalah HTML5 yang dirancang untuk memperbaiki teknologi HTML versi sebelumnya agar dapat mendukung teknologi multimedia terbaru dan tipe isi halaman *web (content)* lainnya. HTML 5 menyediakan elemen-elemen atau *tag* baru yang sebelumnya tidak tersedia dalam HTML versi sebelumnya. Untuk menyatakan (memberitahukan kepada *web browser*) bahwa dokumen HTML yang dibuat adalah dokumen HTML5, maka perlu dituliskan baris kode di bagian paling atas dokumen berupa. Adapun HTML merupakan suatu *form*at data yang digunakan untuk membuat dokumen *hypertext* yang dapat dieksekusi dari satu *platform* komputer ke plat*form* komputer lainya tanpa perlu melakukan suatu perubahan apapun dengan suatu alat tertentu (Junaedi, 2005).

2.5 PHP (*Hypertext Preprocessor*)

PHP singkatan dari *Hypertext Preprocessor* yang digunakan sebagai bahasa *script server-side* dalam pengembangan Web yang disisipkan pada dokumen HTML. (Peranginangin, 2006). PHP adalah bahasa *script server side* yang sengaja dirancang lebih cenderung untuk membuat dan mengembangkan web. Bahasa pemrograman ini memang dirancang untuk para pengembang web agar dapat menciptakan suatu halaman web yang bersifat dinamis. Sekilas tentang PHP, PHP pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995 dan terus dikembangkan hingga saat ini. Ada banyak sekali web termasuk *CMS(Content Menagement System)* yang dibuat menggunakan bahasa PHP, seperti *Wordpress* dan lain-lain (Yudho, 2019).

Pembuatan aplikasi web ataupun web responsif secara lebih mudah dapat dilakukan dengan menggunakan bantuan *Bootstrap* merupakan *Framework* maupun tools yang 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 *user interface* yang cantik seperti *transitions, modal, dropdown, scrollspy, tooltip, tab, pop over, alert, button, carousel,* dan lain-lain. (Juanda, 2008).

2.6 *Cascading Style Sheet*

CSS (*Cascading Style Sheet*) adalah *stylesheet language* yang digunakan untuk mendeskripsikan penyajian dari dokumen yang dibuat dalam *mark up language*. CSS merupakan sebuah dokumen yang berguna untuk melakukan pengaturan pada komponen halaman *web*, inti dari dokumen ini adalah mem*form*at halaman *web* standar menjadi bentuk *web* yang memiliki kualitas yang lebih indah dan menarik Dengan adanya teknologi seperti ini, developer dapat memilah atau memisah antara kode untuk isi halaman web dan kode yang diperlukan khusus untuk menangani tampilan.

Pembuatan aplikasi web ataupun web responsif secara lebih mudah dapat dilakukan dengan menggunakan bantuan *Bootstrap* merupakan *Framework* maupun *tools* yang 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 *user interface* yang cantik seperti *transitions*, *modal*, *dropdown*, *scrollspy*, *tooltip*, *tab*, *pop over*, *alert*, *button*, *carousel*, dan lain-lain. Dengan bantuan *Bootstrap*, *web* responsif dapat dibuat dengan cepat dan mudah dan dapat berjalan sempurna pada *browser-browser* populer seperti *Chrome*, *Firefox* dan *Microsoft Edge* (Juanda, 2008).

2.*7 JavaScript*

JavaScript adalah bahasa *scripting* kecil, ringan, berorientasi objek yang ditempelkan pada kode HTML dan di proses di sisi *client*. *JavaScript* digunakan dalam pembuatan *website* agar lebih interaktif dengan memberikan kemampuan tambahan terhadap HTML melalui eksekusi perintah di sisi browser. *JavaScript* dapat merespon perintah *user* dengan cepat dan menjadikan halaman *web* menjadi responsif. *JavaScript* memiliki struktur sederhana, kodenya dapat disisipkan pada dokumen HTML atau berdiri sebagai satu kesatuan aplikasi.

2.7.1 *Vue.js*

Vue.js adalah *JavaScript Framework* yang dikembangkan untuk membangun antarmuka suatu *software*. *Vue.js* telah menyediakan berbagai macam fungsi *JavaScript* yang telah dimodifikasi sehingga *programmer* dapat lebih mudah untuk membangun *software*, tentunya dengan aturan-aturan tertentu. Selain memudahkan dalam pengembangan, *Vue.js* juga memberi kemudahan kepada pengguna *software* dalam menggunakan *software* itu sendiri dengan *realtime response*, di mana *Vue.js* meminimalkan waktu antara aksi *user* dengan respons perangkat lunak. Vue.js pertama kali dirilis pada Februari 2014 oleh Evan You setelah bekerja di *Google* menggunakan *Angular.js* di beberapa proyek.

Berbeda dengan *Framework* lainnya yang menggunakan prinsip MVC (*Model-View-Controller*). *Vue.js* hanya difokuskan untuk membangun tampilan / hanya bekerja pada *View layer*. Dalam hal lainnya Vue.js juga dapat membantu *programmer* untuk membuat *web* dengan aplikasi *single page* yang canggih dan dapat dikombinasikan dengan *library* browser lainnya. Selain itu *Vue.js* menggunakan *template sintaks* berbasis HTML yang memungkinkan pengguna untuk mendeklarasikan *data*/ *state* ke dalam DOM. Semua *template Vue.js* adalah HTML yang valid yang dapat diuraikan oleh browser sesuai spesifikasi dan parser HTML (Gok dan Khanna, 2013).

2.7.2 JSON

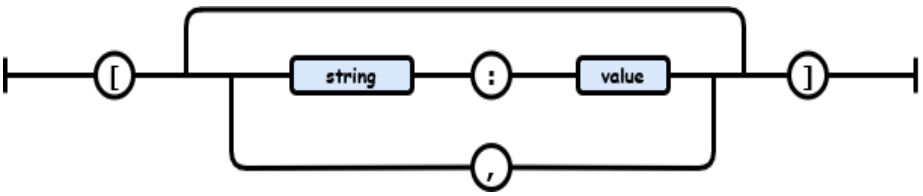
JSON (*JavaScript Object Notation*) adalah *form*at pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan 12 dibuat (*generate*) oleh komputer. *Form*at ini dibuat berdasarkan bagian dari Bahasa *Pemrograman JavaScript*, Standar ECMA-262 Edisi ke-3 - Desember 1999. JSON merupakan *form*at teks yang tidak bergantung pada bahasa pemrograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh *programmer* keluarga C termasuk C, C++, C#, *Java*, *JavaScript*, *Perl*, *Python* dan lain-lain (Scott, 2016). Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran data. JSON terbuat dari dua struktur:

Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), struktur (*struct*), kamus (*dictionary*), Tabel *hash* (*hash table*), daftar berkunci (*keyed list*), atau *associative Array*.

Daftar nilai terurutkan (*an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (*Array*), vektor (*vector*), daftar (*list*), atau urutan (*Sequence*).

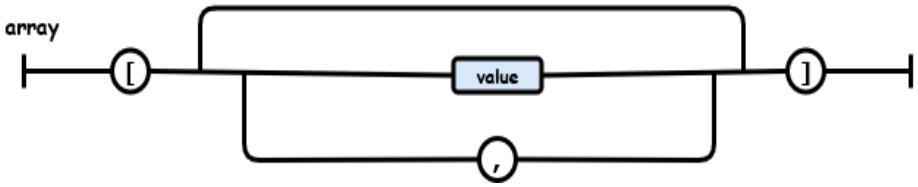
Struktur-struktur data ini disebut sebagai struktur data universal. Pada dasarnya, semua bahasa pemrograman modern mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena *form*at data mudah dipertukarkan dengan bahasa-bahasa pemrograman yang juga berdasarkan pada struktur data ini. JSON menggunakan bentuk sebagai berikut:

1. Objek adalah sepasang nama/nilai yang tidak terurutkan. Objek dimulai dengan { (kurung kurawal buka) dan diakhiri dengan } (kurung kurawal tutup). Setiap nama diikuti dengan : (titik dua) dan setiap pasangan nama/nilai dipisahkan oleh , (koma).(dapat dilihat pada gambar 2.1)



Gambar 2.1 JSON Object

2. Larik adalah kumpulan nilai yang terurutkan. Larik dimulai dengan [(kurung kotak buka) dan diakhiri dengan] (kurung kotak tutup). Setiap nilai dipisahkan oleh , (koma). (dapat dilihat pada gambar 2.2)



Gambar 2.2 JSON Array

2.*8 Framework*

Framework adalah kerangka kerja. *Framework* juga dapat diartikan sebagai kumpulan *script* (terutama *Class* dan *function*) yang dapat membantu *developer/programmer* dalam menangani berbagai masalah-masalah dalam pemrograman. Seperti koneksi ke *database*, pemanggilan variabel, *file*, dan lain-lain sehingga pekerjaan *developer* lebih fokus dan lebih cepat dalam membangun aplikasi.

Framework merupakan komponen pemrograman yang siap digunakan ulang kapan saja sehingga *programmer* tidak harus membuat *script* yang sama untuk tugas yang sama. Misalkan, saat akan membuat aplikasi web berbasis *Ajax* yang setiap kali harus melakukan *XML Http Request* maka *Xajax* telah mempermudah dengan menciptakan sebuah objek khusus yang siap digunakan untuk operasi *Ajax* berbasis *PHP*.

Secara sederhana bisa dijelaskan bahwa *Framework* adalah kumpulan fungsi *(libraries)* sehingga seorang *programmer* tidak perlu lagi membuat fungsi-fungsi dari awal dan biasanya disebut kumpulan *library*. *Programmer* cukup memanggil kumpulan *library* atau fungsi yang sudah ada di dalam *Framework* yang sudah pasti cara menggunakan fungsi-fungsi itu sudah ditentukan sesuai aturan masing-masing.

Beberapa contoh fungsi sekitar yang telah tersedia dalam suatu *Framework* adalah fungsi *paging*, *enkripsi*, *e-mail*, *SEO session*, *security*, kalender, bahasa, grafik tabel bergaya zebra, validasi, *upload*, *captcha*, proteksi terhadap *XSS (XSS filtering)*, *template*, kompresi, XML, dan lain - lain. Contoh dari *Framework PHP* adalah *Laravel*, *Phalcon*, *Slim*, *CakePHP*, *Code Igniter (CI)*, *Symphony*, *Zend*, *Yii*, dan *Kohana*. Sedangkan, *Framework Javascript* juga ada, yakni: *Jquery*, *JSON*, dan *Mootools*. *Framework* untuk *Ruby* adalah *Ruby on Rails (ROR)*. Dengan menggunakan *Framework*, sebuah aplikasi akan disusun secara terstruktur dan rapi karena pasti pembuat *Framework* telah menggunakan *pattern stkitart*, misalnya *MVC* atau sering disebut *Model-View-Controller* yang telah terkenal di kalangan *programmer PHP*. Logikanya seperti ini, jika kita membuat *website* dari nol, kita

akan asal-asalan dalam membuat kode kita sehingga di dalam satu *file* akan terdapat beribu-ribu kode yang sebenarnya dapat kita pisah menjadi beberapa *file* sehingga per*forma* *web* itu tidak menurun.

Jika membuat aplikasi tidak menggunakan *Framework,* maka akan kesulitan sendiri. Misal, saat kita *handling error* pasti kita harus membaca beribu-ribu baris dari kode yang telah kita buat. Padahal kita sudah dikejar waktu untuk membuat fitur lain. Oleh karena itu, *Framework* adalah solusi bagi kita untuk membuat sebuah *software*. (Yudho, 2019)

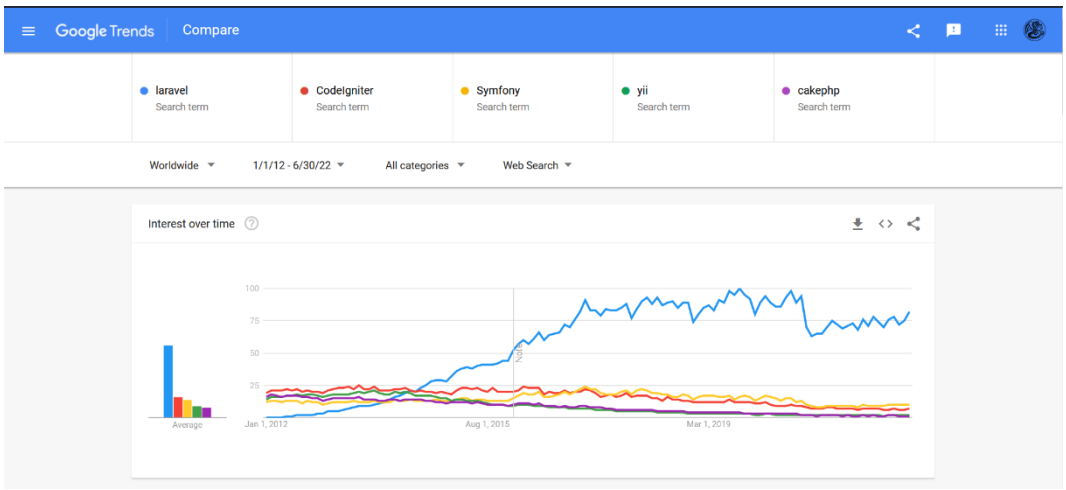
2.*9 Laravel*

Laravel adalah *Framework* berbasis PHP yang sifatnya open *source*, dan menggunakan konsep *Model–View–Controller*. *Laravel* berada di bawah lisensi MIT *License* dengan menggunakan *Github* sebagai tempat berbagi *code* menjalankannya(Naista, 2016). Adapun *Laravel* merupakan salah satu *Framework* PHP yang dikembangkan oleh Taylor Otwell, proyek *Laravel* dimulai pada April 2011. Awal mula, proyek ini dibuat, karena *Otwell* sendiri tidak menemukan *Framework* yang *up-to-date* dengan versi PHP. Mengembangkan *Framework* yang sudah ada juga bukan merupakan ide yang bagus karena keterbatasan sumber daya. Dikarenakan beberapa keterbatasan tersebut, *Otwell* membuat sendiri *Framework* dengan nama *Laravel*. Oleh karena itu, *Laravel* mensyaratkan PHP versi 5.3 ke atas.

Laravel adalah sebuah *Framework* PHP yang dirilis di bawah lisensi MIT dan dibangun dengan konsep MVC (*Model View Controller*). *Laravel* adalah pengembangan *website* berbasis MVC yang ditulis dalam PHP yang dirancang untuk meningkatkan kualitas perangkat lunak dengan mengurangi biaya pengembangan awal dan biaya pemeliharaan, serta untuk meningkatkan pengalaman bekerja dengan aplikasi dengan menyediakan *sintaks* yang ekspresif, jelas, dan menghemat waktu.

Sebagai sebuah *Framework* PHP, *Laravel* hadir sebagai plat*form* web *development* yang bersifat *open source*. Yang menarik dari *Laravel* adalah

sintaksnya ekspresif dan elegan, serta dirancang khusus untuk memudahkan dan mempercepat proses web *development*. Dari data *Google Trend*, menunjukkan bahwa *Laravel* adalah yang paling banyak dicari dan dibaca. (dapat dilihat pada gambar 2.3)



Gambar 2.3 Statistik Pencarian Framework di Google

2.10 *Progressive Web Apps*

Progressive Web Apps (PWA) adalah sebuah istilah untuk aplikasi berbasis web yang menggunakan teknologi web paling mutakhir. PWA sebenarnya hanyalah aplikasi berbasis web biasa, tapi memanfaatkan fitur peramban yang modern agar tampil seolah-olah merupakan aplikasi asli. PWA digambarkan sebagai kumpulan dari teknologi, konsep desain dan WEB API (*Application Programming Interface*) yang bekerja secara bersama untuk memberikan sentuhan aplikasi pada sebuah *mobile web* (Rahul,2016).

Hal ini termasuk berbagai rekomendasi yang tidak spesifik pada desain aplikasi web untuk perangkat *mobile*, seperti preferensi HTTPS melalui HTTP dan desain yang *responsive*. Hal ini juga akan membawa kebutuhan pada API baru untuk peningkatan kualitas pengguna, seperti *Web App Manifest*, *Service Workers* ataupun *Payment Request* API.

Pada Mei 2016 *Google* memperkenalkan *Progressive Web Apps*. *Progressive Web Apps* dirancang oleh Frances Berriman dan *Google Chrome Engineer* Alex Russel. Teknologi dibalik *Progressive Web Apps* adalah *service worker*. Menurut *Google developer* karakteristik dari *Progressive Web Apps* adalah sebagai berikut :

- *Progressive*

  dapat digunakan oleh semua pengguna, terlepas browser apa yang digunakan karena aplikasi dikembangkan secara *progressive*
- *Responsive*

  dapat digunakan pada semua perangkat mulai dari *desktop*, tablet, *smartphone* dan lainnya.
- *Connectivity independent*
- memiliki *service workers* untuk dapat diakses secara *offline* atau pada kualitas koneksi jaringan yang rendah.
- *App-like*

  terasa seperti aplikasi karena *Model* aplikasi *shell* memisahkan fungsi dari konten aplikasi.
- *Fresh*

  selalu *up-to-date* dikarenakan *update* proses dari *service worker*.
- *Safe*

  dilayani via HTTPS untuk mencegah pengintaian dan untuk memastikan konten tidak dirusak.
- *Discoverable*

  yaitu diidentifikasi sebagai "aplikasi" berkat cakupan W3C dan ruang lingkup pendaftaran *service worker*, yang memungkinkan mesin pencari menemukannya.
- *Re-engageable*

  yaitu membuat keterlibatan ulang menjadi mudah dengan adanya fitur seperti *push notification*.
- *Installable*

memungkinkan *user* untuk menambahkan aplikasi ke *home screen* tanpa melalui *appstore*.

- *Linkable*

  dapat berbagi dengan mudah melalui tautan dan tidak memerlukan instalasi.

Progressive *Web Apps* memiliki tiga kebutuhan yang harus dipenuhi, yaitu :

- Harus dilayani melalui *Hypertext Transfer Protocol Secure* (HTTPS).
- Memiliki *web app manifest*.
- Memiliki setidaknya satu *service worker*.

2.10.1 *Web app manifest*

Web app manifest merupakan *file* JSON yang memberitahukan browser tentang aplikasi *web* yang dijalankan seperti bagaimana harusnya berperilaku ketika dilakukan instalasi pada perangkat selular atau *desktop*, ini diperlukan untuk memunculkan permintaan "Tambahkan ke Layar Beranda" (Scott, 2016). *File* manifes mencakup in*form*asi tentang nama aplikasi, ikon yang harus digunakan, *start url* dimana manifes harus dimulai saat diluncurkan, dan banyak lagi (Gaunt & Kinlan 2019).

Saat pengguna menambahkan *website* ke layar beranda, sistem dapat menentukan serangkaian ikon untuk digunakan *browser*. Ikon-ikon ini digunakan di tempat-tempat seperti layar beranda, layar pembuka, dll. Ikon adalah *Array* dari objek gambar. Setiap objek harus menyertakan *src*, *sizes property*, dan tipe gambar.

2.10.2 *Service worker*

Service worker adalah salah satu jenis dari *web worker*, yaitu *script* yang berjalan di belakang *browser* pengguna. *Service worker* pada dasarnya adalah berkas *JavaScript* yang berjalan pada *thread* yang berbeda dengan *main thread browser*, menangani *network request*, *caching*, mengembali-kan *resource* dari *cache*, dan bisa mengirimkan *push message*. Aset *web* dapat disimpan sebagai *cache* lokal, sehingga dengan jaringan internet yang kurang memadai pun,

pengguna tetap mendapat pengalaman yang baik. Aplikasi dapat tetap menjalankan halaman web yang sudah di-*cache* atau memberikan status koneksi tanpa browser menampilkan tulisan *error* karena ketiadaan koneksi internet (Ridho, Pinandhito, & Dewi, 2018).

Langkah-langkah dalam melakukan konfigurasi dasar *service worker* sebagai berikut:

- Daftarkan *service worker* melalui URL (*Uniform Resource Locator*) fungsi *serviceWorkerContainer.register()*.
- Jika berhasil, *service worker* dijalankan di *ServiceWorkerGlobalScope*.
- *Service worker* telah siap untuk memproses *event*.
- Instalasi *service worker* dicoba ketika *service worker* mengontrol halaman yang diakses setelah dan sebelumnya. *Event install* akan selalu dikirim pertama kali ke *service worker*.
- Ketika *handler on install* selesai, *service worker* dipasang.
- Proses aktivasi. Ketika *service worker* terpasang, selanjutnya akan menerima *event activate*. Penggunaan utama dari *onactivate* ini adalah untuk membersihkan sumber daya yang digunakan sebelumnya.
- *Service control* sekarang dapat mengontrol halaman, tapi hanya dibuka setelah *register* telah sukses seperti dokumen mulai aktif dengan atau tanpa *service worker* dan menjaganya selama masih digunakan. Jadi dokumen harus dimuat ulang agar benar-benar terkontrol.

2.11 *Database*

Basis data (*database*) diartikan sebagai markas atau gudang data, tempat bersarang atau berkumpul data. Prinsip utama basis data adalah pengaturan data dengan tujuan utama fleksibilitas dan kecepatan dalam pengambilan data kembali. Adapun tujuan basis data diantaranya sebagai efisiensi yang meliputi *speed*, *space*

dan *Accurancy*, menangani data dalam jumlah besar, kebersamaan pemakaian, dan meniadakan duplikasi (Yakub,2012).

MySQL adalah salah satu jenis *database server* yang sangat terkenal dan banyak digunakan untuk membangun aplikasi web yang menggunakan *database* sebagai sumber dan pengolahan datanya. (Arief,2011) *MySQL* yang merupakan singkatan dari *"My Structured Query Language"* adalah *database* yang paling favorit saat ini. Program ini berjalan sebagai *server* yang menyediakan *multi-user*, mengakses ke sejumlah *database* baik *multi-thread* maupun *multi-user*, dan telah *diinstal* oleh sekitar enam juta kali di seluruh dunia. *MySQLAB* gratis, di bawah lisensi *GNU General Public License (GPL)*, tetapi ada juga *MySQL* yang berbayar. (Yudho, 2019)

Kelebihan *MySQL*:

- *Free*, stabil, dan tangguh.
- Fleksibel dengan berbagai pemrograman.
- *Security* atau keamanan yang baik.
- Dukungan dari banyak komunitas.
- Kemudahan *management database*.
- Mendukung transaksi.
- Perkembangan *software* cukup cepat.

Kekurangan *MySQL*:

- Kurang mendukung koneksi ke bahasa pemrograman visual seperti VB, *Delphi*, dan *Foxpro* dikarenakan koneksi ini menyebabkan *field* yang dibaca harus sesuai dengan koneksi dari program visual tersebut.
- Data yang ditangani belum begitu besar.
- Lambat untuk *query* yang kompleks, seperti LEFT JOIN yang banyak dan penggunaan *SubQuery*.
- Belum mendukung *Windowing Function*.

2.12 *Unified Modelling Language* (UML)

UML merupakan bahasa visual untuk pe*Model*an dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung (Rosa & Shalahuddin, 2013). Jadi, dapat disimpulkan UML merupakan Penyampaian Karakteristik sistem in*form*asi yang dirumuskan dalam bentuk gambar atau penjelasan lainnya yang bertujuan untuk memudahkan dalam memahami. Untuk setiap pendekatan bahasa visual untuk pe*Model*an memiliki kelebihan dan kekurangan yang berbeda-beda, tetapi UML memiliki enam keunggulan utama sebagai berikut:

- Bahasa *form*al

  Setiap objek elemen dalam bahasa ini memiliki arti yang sangat jelas, sehingga anda yakin dapat me*Model*kan aspek tertentu dari sistem. Tanpa ragu untuk tidak dipahami.
- Bahasa yang singkat

  Seluruh bahasa terdiri dari notasi sederhana dan mudah.
- Bahasa yang komprehensif

  Dapat menggambarkan semua aspek penting dalam sebuah sistem.
- Bahasa yang terukur

  Jika diperlukan, bahasa ini cukup *form*al untuk menangani proyek yang bersifat *massive*, namun dapat digunakan juga digunakan dalam proyek skala kecil untuk menghindari kerja keras.
- Bahasa yang dibangun dari pelajaran yang didapat

  UML merupakan kumulasi dari praktik dalam komunitas berorientasi objek selama 15 tahun terakhir.
- Bahasa standar

  UML dikontrol oleh *grub* terstandarisasi yang bersifat terbuka dengan kontribusi aktif dari penyedia layanan UML dan akademisi di seluruh dunia yang menolak bersifat "*vendor lock-in*". Standarisasi ini

memastikan kemampuan trans*form*asi dan *interopabilitas* UML, yang berarti anda tidak terikat pada produk tertentu.

2.12.1 *UseCase diagram*

Use case diagram adalah satu dari berbagai jenis diagram UML (*Unified Modelling Language*) yang menggambarkan hubungan interaksi antara sistem dan aktor. *Use Case* dapat mendeskripsikan tipe interaksi antara si pengguna sistem dengan sistemnya. Langkah awal untuk melakukan pe*Model*an, tentu perlunya suatu diagram yang mampu menjabarkan aksi aktor dengan aksi sistem itu sendiri, seperti yang terdapat pada *use case diagram*. *Use case* adalah komponen gambaran fungsional dalam sebuah sistem. Sehingga konsumen maupun pembuat saling mengenal dan mengerti mengenai alur sistem yang akan dibuat. (Julianto, 2021)

Adapun, fungsi dari *use case diagram* sebagai berikut:

- Berguna memperlihatkan proses aktivitas secara urut dalam sistem.
- Mampu menggambarkan proses bisnis, bahkan menampilkan urutan aktivitas pada sebuah proses.
- Sebagai *bridge* atau jembatan antara pembuat dengan konsumen untuk mendeskripsikan sebuah sistem.

Manfaat dari *use case diagram* di antaranya:

- Menggunakannya sebagai kebutuhan verifikasi.
- Menjadi gambaran *interface* dari sebuah sistem karena setiap sistem yang dibangun haruslah memiliki *interface*.
- Mengidentifikasi siapa saja orang yang dapat berinteraksi dengan sistem, serta apa yang dapat dilakukan oleh sistem.
- Memberikan kepastian mengenai kebutuhan sistem, sehingga tidak membingungkan.
- Memudahkan proses komunikasi antara domain *expert* dan *end user*.

Adapun Komponen-komponen pada *use case diagram* di antaranya sebagai berikut.

- Sistem

  Sebuah sistem digambarkan ke dalam bentuk persegi, fungsinya untuk membatasi use case dengan interaksi dari luar sistem. Sistem pada umumnya diberikan label yang sesuai. Namun, umumnya sistem ini tidaklah diberi gambar karena kita tidak terlalu memberikan makna pada sebuah diagram.

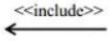
- *Actor*

  Banyak yang berspekulasi bahwa *actor* adalah bagian dari diagram. Padahal apabila kita mencari in*form*asi lebih dalam mengenai soal ini, ternyata *actor* bukanlah bagian dari diagram, Peran *actor* sangat penting, tentunya menciptakan *use case* jadi lebih mudah. Fungsi *Actor* menjelaskan siapa yang berinteraksi dengan sistem. *Actor* akan memberikan in*form*asi kepada sistem, serta menerima in*form*asi dari sistem.

  Keduanya bisa terjadi secara bersamaan. Aktor tidak memberikan kontrol terhadap sistem, namun hanya memberikan gambaran mengenai hubungannya dengan sistem. Ternyata, inilah beberapa alasan mengapa *actor* dapat berhubungan dengan sistem lain:

  a. Jika terdapat relasi sistem lain dengan sistem yang sedang dibuat.
  b. Terdapat eksternal *resource* yang digunakan oleh sistem.
  c. Adanya kepentingan terhadap sistem, yaitu alur in*form*asi baik penerima maupun arus sistem saling memiliki kepentingan.
  d. Terdapat seseorang atau pihak lain yang akan mengelola sistem.

Adapun simbol-simbol dari komponen *use case diagram* dapat dilihat dalam gambar 2.4 berikut:

Simbol	Keterangan
	Aktor : Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan *use case*
	*Use case* : Abstraksi dan interaksi antara sistem dan aktor
	Association : Abstraksi dari penghubung antara aktor dengan use case
	Generalisasi : Menunjukkan spesialisasi aktor untuk dapat berpartisipasi dengan use case
	Menunjukkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya
	Menunjukkan bahwa suatu use case merupakan tambahan fungsional dari use case lainnya jika suatu kondisi terpenuhi

Gambar 2.4 komponen UseCase Diagram

2.12.2 *Class diagram*

Class diagram atau diagram kelas adalah salah satu jenis diagram struktur pada UML yang menggambarkan dengan jelas struktur serta deskripsi *Class*, atribut, metode, dan hubungan dari setiap objek. Ia bersifat statis, dalam artian diagram kelas bukan menjelaskan apa yang terjadi jika kelas-kelasnya berhubungan, melainkan menjelaskan hubungan apa yang terjadi. Diagram kelas ini sesuai jika diimplementasikan ke proyek yang menggunakan konsep *object-oriented* karena gambaran dari *Class diagram* cukup mudah untuk digunakan. Desain *Model* dari diagram kelas ini sendiri dibagi menjadi dua bagian. Bagian pertama merupakan penjabaran dari *database*. Bagian kedua merupakan bagian dari modul MVC, yang memiliki *Class interface*, *Class control*, dan *Class entity*.

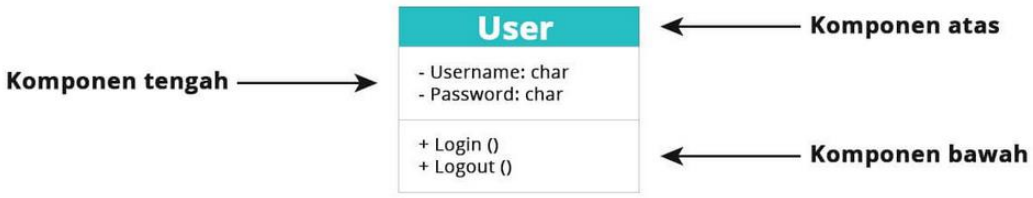
Diagram kelas ini memiliki beberapa fungsi, fungsi utamanya yaitu menggambarkan struktur dari sebuah sistem. Berikut ini adalah fungsi-fungsi lainnya:

- Menunjukkan struktur dari suatu sistem dengan jelas.
- Meningkatkan pemahaman tentang gambaran umum atau skema dari suatu program.
- Dapat digunakan untuk analisis bisnis dan digunakan untuk membuat *Model* sistem dari sisi bisnis.
- Dapat memberikan gambaran mengenai sistem atau perangkat lunak serta relasi-relasi yang terkandung di dalamnya.

Menggunakan diagram kelas memberikan banyak keunggulan bagi proses pengembangan perangkat lunak dan dalam bisnis. Berikut ini adalah keunggulan dari diagram kelas:

- Diagram kelas berfungsi untuk menjelaskan suatu *Model* data untuk sebuah program, baik *Model* data sederhana maupun kompleks.
- Memberikan gambaran umum tentang skema aplikasi dengan jelas dan lebih baik.
- Membantu kamu untuk menyampaikan kebutuhan dari suatu sistem.

Diagram kelas memiliki tiga komponen penyusun (dapat dilihat pada gambar 2.5). Berikut ini adalah komponen-komponennya:



Gambar 2.5 komponen Class Diagram

- **Komponen atas**

  Komponen ini berisikan nama *Class*. Setiap *Class* pasti memiliki nama yang berbeda-beda, sebutan lain untuk nama ini adalah *simple name* (nama sederhana).

- **Komponen tengah**

  Komponen ini berisikan atribut dari *Class*, komponen ini digunakan untuk menjelaskan kualitas dari suatu kelas. Atribut ini dapat

menjelaskan dapat ditulis lebih detail, dengan cara memasukkan tipe nilai.

- **Komponen bawah**

  Komponen ini menyertakan operasi yang ditampilkan dalam bentuk daftar. Operasi ini dapat menggambarkan bagaimana suatu *Class* dapat berinteraksi dengan data.

Setelah mengetahui penjelasan tentang diagram kelas, sekarang akan membahas hubungan antar kelasnya. Ada tiga hubungan dalam diagram kelas. Berikut ini adalah penjelasannya:

- **Asosiasi**

  Pertama ada asosiasi. Asosiasi dapat diartikan sebagai hubungan antara dua *Class* yang bersifat statis. Biasanya asosiasi menjelaskan *Class* yang memiliki atribut tambahan seperti *Class* lain.

- **Agregasi**

  Agregasi adalah hubungan antara dua *Class* di mana salah satu *Class* merupakan bagian dari *Class* lain, tetapi dua *Class* ini dapat berdiri masing-masing.

- **Pewarisan**

  Pewarisan atau *inheritance* dapat disebut juga *generalization* dalam *Class* diagram adalah suatu kemampuan untuk mewarisi seluruh atribut dan metode dari *Class* asalnya (*superClass*) ke *Class* lain (*subClass*). (setiawan,2021)

2.12.3 ER*(Entity relationship) diagram*

ERD (*Entity relationship Diagram*) atau diagram hubungan entitas adalah sebuah diagram yang digunakan untuk perancangan suatu *database* dan menunjukkan relasi atau hubungan antar objek atau entitas beserta atribut-atributnya secara detail. Dengan menggunakan ERD, sistem *database* yang sedang dibentuk dapat digambarkan dengan lebih terstruktur dan terlihat rapi. Selain digunakan dalam perancangan *database*, ERD sendiri sering digunakan untuk

debugging database jika terjadi masalah pada *database*. Untuk melakukan debug pada *database* bukanlah hal yang mudah, terlebih lagi jika *database* yang mengalami masalah memiliki banyak tabel dan memerlukan penulisan SQL yang kompleks. Dengan menggambarkan skema *database* menggunakan ERD, kamu menjadi lebih mudah untuk menemukan permasalahan yang terjadi dalam *database* dan menyelesaikan masalah dengan mudah.

Sebelum membuat perancangan sistem yang tepat, harus terlebih dahulu mengetahui jenis *Model* data yang digunakan. Karena *Model* data tersebut nantinya akan berpengaruh dalam pengembangan sistem. *Model* ini juga berguna untuk membuat dokumentasi dari segala bentuk arsitektur data. *Model* ini dibagi ke dalam tiga *Model*. Berikut adalah penjelasannya.

- ***Model* data konseptual**

  Model data ini adalah *Model* data paling tinggi karena di dalamnya berisi data-data yang detail. Data konseptual ini dapat kamu gunakan sebagai dasar untuk membuat satu atau lebih *Model* data logis. Tujuan dari pengembangan *Model* data konseptual adalah untuk memberikan gambaran yang jelas mengenai struktur *database* yang terdiri dari entitas dan relasi antara setiap entitas.

- ***Model* data logis**

  Berikutnya adalah *Model* data logis. *Model* data logis ini adalah pengembangan dari *Model* data konseptual, itu sebabnya dalam proses pembuatannya *Model* data ini dibuat lebih rinci dari *Model* data konseptual dan dibuat setelah *Model* data konseptual selesai dibuat. *Model* ini digunakan untuk menambahkan in*form*asi secara eksplisit ke dalam unsur-unsur *Model* konseptual. Terdapat juga beberapa komponen dalam *Model* data ini, seperti entitas data master, operasional, dan transaksional.

- ***Model* data fisik**

  Yang terakhir adalah *Model* data fisik. *Model* data fisik adalah pengembangan dari masing-masing *Model* data logis. *Model* data ini biasanya digunakan untuk merancang sebuah *database*.

Setelah mengetahui apa saja *Model* data dari ER diagram, sekarang akan dibahas komponen-komponen yang digunakan dalam membuat ER diagram. Dalam sebuah ERD sendiri terdapat empat komponen utama untuk me*Model*kan suatu sistem. Berikut adalah komponen-komponennya.

- **Entitas**

  Yang pertama adalah entitas. Entitas merupakan sekumpulan objek yang dapat diidentifikasi secara unik dan berbeda satu dengan yang lainnya. Entitas ini biasanya digambarkan dengan lambang persegi panjang.

  Lalu, ada juga yang dinamakan "Entitas lemah". Entitas lemah ini digambarkan dengan lambang persegi panjang kecil di dalam persegi panjang yang lebih besar. Mengapa disebut dengan entitas lemah? Karena entitas tersebut harus terhubung langsung dengan entitas lain, sebab entitas lemah ini tidak dapat diidentifikasi secara unik.

- **Atribut**

  Selanjutnya adalah atribut. Setiap entitas pasti memiliki atribut yang berfungsi untuk menjelaskan atau mendeskripsikan karakteristik dari entitas tersebut. Ada beberapa jenis atribut yang biasa digunakan dalam ERD. Berikut adalah jenis-jenisnya.

  1. **Atribut kunci**

     Atribut kunci atau *Key Attributes* adalah atribut yang berfungsi untuk menentukan data yang bersifat penting. Biasanya atribut kunci ini berbentuk angka atau numerik. Contoh dari atribut ini adalah No. KTP, NIM (Nomor Induk Mahasiswa), dan lain-lain. Atribut kunci ini dilambangkan dengan lingkaran lonjong dengan keterangan di dalamnya yang diberi garis bawah.

2. **Atribut simpel**

   Berikutnya adalah atribut simpel. Atribut simpel adalah atribut yang tidak dapat dipecah lagi dan bernilai tunggal. Contoh dari atribut ini adalah alamat kantor, nama penerbit, dan lain-lain.

3. **Atribut multinilai**

   Atribut multinilai atau *Multivalue Attributes* adalah atribut yang memiliki atribut lebih dari satu nilai. Contoh dari atribut ini adalah sebuah website artikel yang memiliki beberapa penulis.

4. **Atribut gabungan**

   Selanjutnya adalah atribut gabungan atau *Composite Attributes*. Atribut gabungan adalah atribut yang terdiri dari beberapa atribut yang berukuran lebih kecil dan memiliki arti tertentu. Contoh dari atribut ini adalah sebuah nama yang terdiri atas nama depan, nama tengah, dan nama belakang.

5. **Atribut derivatif**

   Yang terakhir adalah atribut derivatif. Atribut derivatif adalah atribut yang dihasilkan dari atribut lain dan atributnya tidak wajib untuk ditulis dalam *Entity relationship Diagram*. Contoh dari atribut ini adalah selisih harga, usia, dan kelas.

- **Relasi**

Komponen ketiga adalah relasi atau *relation*. Relasi dalam ERD adalah hubungan yang terjadi antara satu atau lebih entitas. Relasi sendiri sering disebut dengan proses. Komponen ini digambarkan dengan lambang belah ketupat. Terdapat tiga jenis relasi yang digunakan dalam ERD dan perlu kamu ketahui, berikut adalah jenisnya.

1. ***One to one***

   One to one berarti setiap entitas hanya dapat memiliki relasi dengan satu entitas lain. Contohnya seperti data mahasiswa dengan NIM (Nomor Induk Siswa).

2. ***One to many***

   One to many memiliki arti satu entitas dapat memiliki relasi dengan beberapa entitas, begitu pula sebaliknya. Contoh dari implementasi one to many ini adalah jurusan dengan mahasiswanya.

3. ***Many to many***

   Many to many memiliki arti setiap entitas yang ada dapat memiliki relasi dengan entitas lain, begitu pula sebaliknya. Contoh dari relasi ini adalah mahasiswa dengan data terkait UKM (Unit Kegiatan Mahasiswa).

- **Garis**

  Komponen terakhir adalah garis. Dalam ERD sendiri garis digunakan untuk menunjukkan hubungan entitas dalam ERD. Selain menjadi penghubung, garis juga dapat menunjukkan alur atau *flow* dari suatu ERD. (setiawan,2021)

2.12.4 *Activity Diagram*

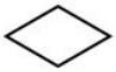
Activity diagram, dalam bahasa Indonesia diagram aktivitas, yaitu diagram yang dapat me*Model*kan proses-proses yang terjadi pada sebuah sistem. Runtutan proses dari suatu sistem digambarkan secara vertikal. *Activity diagram* merupakan pengembangan dari *Use Case* yang memiliki alur aktivitas. Alur atau aktivitas berupa bisa berupa runtutan menu-menu atau proses bisnis yang terdapat di dalam sistem tersebut. Dalam buku "**Rekayasa Perangkat Lunak"** karangan Rosa A.S mengatakan, "Diagram aktivitas tidak menjelaskan kelakuan aktor. Dapat diartikan bahwa dalam pembuatan *Activity diagram* hanya dapat dipakai untuk menggambarkan alur kerja atau aktivitas sistem saja."

Berikut beberapa tujuan dari *Activity diagram:*

- Menjelaskan urutan aktivitas dalam suatu proses.
- Di dalam dunia bisnis biasanya digunakan untuk *Modeling* (memperlihatkan urutan proses bisnis).
- Mudah dalam memahami proses yang ada dalam sistem secara keseluruhan.

- Merupakan metode perancangan yang terstruktur, mirip dengan *Flowchart* maupun *Data Flow Diagram* (DFD).
- Mengetahui aktivitas aktor/pengguna berdasarkan *use case*/diagram yang dibuat sebelumnya.

Berikut (pada gambar 2.6) ini merupakan komponen-komponen dalam *Activity diagram*:

Simbol	Nama	Keterangan
	Status awal	Sebuah diagram aktivitas memiliki sebuah status awal.
	Aktivitas	Aktivitas yang dilakukan sistem,aktivitas biasanya diawali dengan kata kerja.
	Percabangan / Decision	Percabangan dimana ada pilihan aktivitas yang lebih dari satu.
	Penggabungan / Join	Penggabungan dimana yang mana lebih dari satu aktivitas lalu digabungkan jadi satu.
	Status Akhir	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
	Swimlane	Swimlane memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

Gambar 2.6 Komponen Activity Diagram

Berikut penjelasan lengkapnya mengenai komponen-komponen pada *Activity diagram* di atas :

- ***Start Point* atau *Initial State* (Titik Mulai/Status Awal)**

  Start Point adalah lingkaran hitam kecil. Biasanya digunakan untuk menandakan status awal, tindakan awal, atau titik awal aktivitas untuk setiap *Activity diagram*.

- ***Activity* (Aktivitas)**

  Activity merupakan aktivitas yang dilakukan atau sedang terjadi dalam sistem. Biasanya diawali dengan "kata kerja" dari aktivitas yang dilakukan.

- ***Decision* atau Percabangan**

  Percabangan atau *decision* merupakan suatu titik atau *point* yang mengindikasikan suatu kondisi di mana adanya kemungkinan dalam perbedaan transisi. Hal tersebut diperlukan ketika sistem yang dimiliki memiliki beberapa kemungkinan atau jalan alternatif.

- ***Synchronization***

  Synchronization dibagi menjadi 2 bagian, yaitu *fork* dan *join*.

  1. *Fork* (percabangan) digunakan untuk memecah *behaviour* (tingkah laku) menjadi *Activity* atau *action* (aksi) secara paralel.
  2. *Join* (penggabungan) digunakan untuk menghubungkan kembali *Activity* dengan *action* secara paralel.

- ***Merge***

  Menggabungkan *flow* yang sudah dipecah menjadi beberapa bagian oleh suatu *flow*.

- ***Swimlanes***

  Memecah *Activity diagram* menjadi kolom dan baris untuk membagi tanggung jawab objek-objek yang melakukan suatu aktivitas.

- ***Transition***

  Digunakan untuk menunjukkan aktivitas selanjutnya dan sebelumnya.

- **Notasi akhir (*end state*)**

  Notasi akhir digunakan untuk menandakan proses tersebut berakhir. Pada UML, notasi akhir dapat digambarkan dengan simbol sebuah *bull's eye* (mata sapi).

Terkadang menggunakan percabangan (*decision*) dengan *fork* adalah hal yang keliru. Sebab *Decision* digunakan untuk memecah aktivitas yang bersifat kondisional. Contohnya pilihan Ya atau Tidak, jika opsi Ya**,** maka terjadi aksi baru

dan jika Tidak**,** maka menolak aksi baru. Sedangkan *fork* digunakan untuk memecah *behaviour* menjadi aktivitas yang paralel, contohnya seperti pengguna dapat memilih, menambah, mengubah, serta bisa juga menghapus. (Juliarto,2021)

2.12.5 *Sequence Diagram*

Sequence diagram atau diagram urutan adalah sebuah diagram yang digunakan untuk menjelaskan dan menampilkan interaksi antar objek-objek dalam sebuah sistem secara terperinci. Selain itu *Sequence diagram* juga akan menampilkan pesan atau perintah yang dikirim, beserta waktu pelaksanaannya. Objek-objek yang berhubungan dengan berjalannya proses operasi biasanya diurutkan dari kiri ke kanan.

diagram ini terdiri dari dua dimensi, yaitu dimensi vertikal yang menunjukkan waktu dan dimensi horizontal yang menunjukkan objek-objek. Tiap-tiap objek, termasuk *actor*, memiliki waktu aktif yang digambarkan dengan kolom vertikal yang disebut dengan *lifeline*. Sementara itu, pesan atau perintah digambarkan sebagai garis panah dari satu *lifeline* ke *lifeline* yang lain. *Sequence Diagram* dapat digunakan untuk menggambarkan serangkaian langkah yang dilakukan sebagai respon dari sebuah peristiwa untuk menghasilkan suatu *output* tertentu.

tujuan utama dari pembuatan diagram urutan adalah untuk mengetahui urutan kejadian yang dapat menghasilkan *output* yang diinginkan. Selain itu, tujuan dari diagram urutan ini mirip dengan *Activity* diagram loh, seperti menggambarkan alur kerja dari sebuah aktivitas, serta dapat menggambarkan aliran data dengan lebih detail, termasuk data atau perilaku yang diterima atau dikirimkan.

Berikut beberapa komponen utama yang sering digunakan:

- **Aktor**

  Komponen yang pertama adalah aktor. Komponen ini menggambarkan seorang pengguna (*user*) yang berada di luar sistem dan sedang berinteraksi dengan sistem. Dalam *Sequence diagram*, aktor biasanya digambarkan dengan simbol *stick figure*.

- **Activation box**

  Selanjutnya ada *activation box*. Komponen *activation box* ini merepresentasikan waktu yang dibutuhkan suatu objek untuk menyelesaikan tugasnya. Semakin lama waktu yang diperlukan, maka secara otomatis *activation box*-nya juga akan menjadi lebih panjang. Komponen ini digambarkan dengan bentuk persegi panjang.

- ***Lifeline***

  lifeline adalah Komponen yang digambarkan dengan bentuk garis putus-putus. *Lifeline* ini biasanya memiliki kotak yang berisi objek yang memiliki fungsi untuk menggambarkan aktivitas dari objek.

- **Objek**

  objek adalah Komponen yang digambarkan memiliki bentuk kotak yang berisikan nama dari objek dengan garis bawah. Biasanya objek berfungsi untuk mendokumentasikan perilaku sebuah objek pada sebuah sistem.

- ***Messages***

  messages atau pesan adalah Komponen untuk menggambarkan komunikasi antar objek. *Messages* biasanya muncul secara berurutan pada *lifeline*. Komponen *messages* ini direpresentasikan dengan anak panah. Inti dari sebuah diagram urutan terdapat pada komponen *lifeline* dan *messages* ini.(setiawan,2021)

2.12.6 *Deployment Diagram*

Deployment diagram adalah sebuah diagram yang terdiri dari beberapa bentuk UML. Salah satunya Kotak tiga dimensi, yang dikenal sebagai node, mewakili elemen perangkat lunak atau perangkat keras utama, atau node, dalam sistem. Garis dari node ke node menunjukkan hubungan, dan bentuk yang lebih kecil yang terdapat di dalam kotak mewakili artefak perangkat lunak yang digunakan.

Berikut beberapa komponen utama yang sering digunakan dalam *Deployment diagram*:

- *Artefact*

  artefak merupakan Produk yang dikembangkan oleh perangkat lunak, dilambangkan dengan persegi panjang dengan nama dan kata "artefak" yang diapit oleh panah ganda.
- Asosiasi (hubungan)

  Garis yang menunjukkan pesan atau jenis komunikasi lain antara node. Komponen: Persegi panjang dengan dua tab yang menunjukkan elemen perangkat lunak.
- *Dependency*

  Garis putus-putus yang berakhir dengan panah, yang menunjukkan bahwa satu node atau komponen bergantung pada yang lain.
- *Interface*

  Lingkaran yang menunjukkan hubungan kontraktual. Objek-objek yang mewujudkan antarmuka harus menyelesaikan semacam kewajiban.
- *Node*

  Objek perangkat keras atau perangkat lunak, ditunjukkan oleh kotak tiga dimensi.
- *Node as container*

  Sebuah node yang berisi *node* lain di dalamnya di mana *node* berisi komponen.
- *Stereotype*

  Perangkat yang terdapat di dalam *node*, disajikan di bagian atas *node*, dengan nama yang diapit oleh panah ganda.

2.13 *Useability*

Usability berasal dari kata *usable* yang secara umum berarti dapat digunakan dengan baik. Sesuatu dapat dikatakan berguna dengan baik apabila kegagalan dalam penggunaannya dapat dihilangkan atau diminimalkan serta memberi manfaat dan kepuasan kepada pengguna [2]. Dalam interaksi antara manusia dengan komputer, Usabilitas atau juga disebut “ketergunaan” berkaitan

dengan kemudahan dan keterbacaan in*form*asi sekaligus pengalaman navigasi yang *user-friendly*. Pembahasan mengenai *interface* (antarmuka) yang *user-friendly* biasanya digunakan untuk halaman *website* atau perangkat lunak (*software*) agar dapat digunakan secara lebih efisien, mudah, dan memberikan pengalaman yang menyenangkan.

Usability merupakan ukuran kualitas pengalaman pengguna saat berinteraksi dengan suatu produk atau sistem berupa aplikasi perangkat lunak, teknologi bergerak, situs web, maupun peralatan lainnya (Handiwidjojo, Lussy: 2016). Sementara menurut *International Organizationfor Standardization* (ISO), *usability* mengukur sejauh mana produk dapat digunakanoleh pengguna untuk mencapai tujuan yang diharapkan meliputi efektivitas, efisiensi, dan kepuasan.(Barnum:2011)

2.14 Mengukur *Useablity* menggunakan *USE Questionnaire.*

Kuesioner ini dibangun menggunakan tujuh poin skala penilaian pengguna diminta untuk menilai setuju dengan pernyataan, mulai dari sangat setuju sampai sangat setuju. Berbagai bentuk kuesioner digunakan untuk mengevaluasi sikap pengguna terhadap berbagai produk. Faktor analisis berikut menyarankan bahwa pengguna mengevaluasi produk menggunakan tiga dimensi, kegunaan, kepuasan, dan kemudahan penggunaan.

Pengaruh dari dimensi lainnya ditemukan, tetapi ketiga dimensi tadi berfungsi membedakan ketiganya paling efektif. Korelasi parsialnya terhitung menggunakan skala yang diturunkan untuk dimensi yang menunjukkan penggunaan dan kegunaan mempengaruhi seseorang yang lain, sehingga peningkatan kemudahan penggunaan meningkatkan peringkat kegunaan dan begitupun sebaliknya. Ketika keduanya mendorong kepuasan, kegunaan yang relatif kurang penting ketika sistem internal yang harus digunakan pengguna. Pengguna lebih bervariasi dalam memberi peringkat kegunaan mereka.

Seperti yang diharapkan dari literatur, kepuasan sangat terkait dengan penggunaan (aktual maupun prediksi). Untuk sistem internal, item yang berkontribusi pada kemudahan penggunaan untuk produk lainnya sebenarnya

dapat dipisahkan menjadi dua faktor, kemudahan mempelajari dan kemudahan penggunaan (yang sangat jelas sangat berkorelasi) item yang muncul di seluruh tes untuk tiga faktor ditambah kemudahan pembelajaran tercantum di bawah ini. Butir-butir soal yang dicetak miring memuat faktor-faktor tersebut secara relatif bersifat kurang kuat.

Tabel 2.1 Mengukur *useability (usefullness* dan *Ease of use)*

NO	Usefullness	NO	Ease of use
1	Aplikasi ini membantu saya bekerja lebih efektif	9	Aplikasi ini mudah untuk digunakan
2	Aplikasi ini membantu saya menjadi lebih produktif	10	Aplikasi ini tidak untuk digunakan
3	Aplikasi berguna bagi saya	11	Aplikasi ini sangat friendly
4	Aplikasi ini mememberikan saya kemudahan dalam mengatur aktivitas perpustakaan	12	Aplikasi ini memerlukan langkah yang sedikit untuk menyelesaikan keperluan saya
5	Aplikasi ini membantu saya menyelesaikan pekerjaan	13	Aplikasi ini flexibel
6	Aplikasi ini menghemat waktu saya	14	Aplikasi ini cukup interaktif
7	Aplikasi ini memenuhi kebutuhan saya	15	Saya dapat menggunakan aplikasi ini tanpa memakai buku panduan
8	Aplikasi ini sesuai dengan harapan saya	16	Saya tidak mendapatkan adanya ketidakkonsistenan pada aplikasi ini
		17	Baik pengunjung dan pengguna reguler akan menyukai aplikasi ini
		18	Saya dapat memperbaiki kesalahan dengan cepat dan mudah
		19	Saya dapat menggunakan aplikasi dengan tepat setiap saat

Pada Tabel 2.1 dapat dilihat poin-poin pertanyaan yang akan dibetikan kepada kuesioner yang melingkupi cakupan kegunaan (*usefullness*), dan kemudahan dalam menggunakan (*ease of use*).

Tabel 2.2 Mengukur *useability (ease of learning* dan *satisfaction)*

NO	Ease of learning	NO	Satisfaction
20	Saya dapat menguasai aplikasi dengan cepat	24	saya puas dengan aplikasi ini
21	Saya dengan mudah mengingat cara menggunakan aplikasi ini	25	saya akan merekomendasikan aplikasi ini ke kerabat
22	Aplikasi ini dapat di kuasai dengan mudah	2	menyenangkan menggunakan aplikasi ini
23	Saya dengan cepat menjadi terampil menggunakan aplikasi ini	27	aplikasi ini berkerja sesuai dengan yang saya inginkan
		28	aplikasi ini bagus
		29	saya merasa membutuhkan aplikasi ini
		30	aplikasi ini menarik untuk digunakan

Pada Tabel 2.2 dapat dilihat poin-poin pertanyaan yang akan dibetikan kepada kuesioner yang melingkupi cakupan Kemudahan dalam menguasai (*Ease of learning*), dan Kepuasan (*satisfaction*).

Hal ini bekerja untuk memperbaiki item dan pertimbangan yang lebih lanjut. Ada beberapa bukti bahwa situs web dan produk konsumen tertentu memiliki dimensi tambahan kesenangan atau estetika terkait dengan pembuatan produk yang menarik. Namun untuk variabel dependen yang menjadi minat utama, item-item ini tampaknya cukup kuat. Bentuk kuesioner yang pendek mudah dibuat dengan menggunakan tiga atau empat item yang paling berbobot untuk setiap faktor.