**RANCANG BANGUN *APLIKASI* INVENTARISASI ASET BERBASIS *WEBSITE* PADA DESA KABUNDERAN**

**Skripsi**

****

Disusun oleh

**Satria Fachreza Akbar**

**16.11.0231**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS AMIKOM PURWOKERTO**

**PURWOKERTO**

**2020**

**BAB II**

**TINJAUAN PUSTAKA**

1. **Landasan Teori**
2. **Rancang Bangun**

Rancang Bangun adalah menciptakan dan membuat suatu *aplikasi* ataupun sistem yang belum ada pada suatu instansi atau objek tersebut (Prasetyo, 2015). Menurut (Lestariningsih dkk, 2015) rancang bangun merupakan kegiatan menerjemahkan hasil analisa ke dalam bentuk paket perangkat lunak kemudian menciptakan suatu sistem tersebut ataupun memperbaiki sistem yang sudah ada. Rancang Bangun adalah suatu istilah umum untuk membuat atau mendesain suatu objek dari awal pembuatan sampai akhir pembuatan (Fajriyah dkk , 2017).

Dari pengertian di atas dapat disimpulkan bahwa rancang bangun adalah proses pembuatan sistem yang belum ada pada suatu objek yang di desain dari awal pembuatan sampai akhir pembuatan.

1. ***Aplikasi***

*Aplikasi* adalah kumpulan perintah program yang dibuat untuk melakukan pekerjaan-pekerjaan (Utari dkk, 2016). *Aplikasi* adalah perangkat lunak yang dibuat untuk melayani kebutuhan akan beberapa aktivitas dan mengolah data menjadi suatu informasi yang berguna bagi masyarakat. Dengan menempatkan *aplikasi* dalam sebuah server maka *aplikasi* dapat diakses kapan saja dan dimana saja (Febrianto dkk, 2017). *Aplikasi* adalah program siap pakai yang dapat digunakan untuk menjalankan perintah-perintah dari pengguna *aplikasi* tersebut dengan tujuan mendapatkan hasil yang lebih akurat sesuai dengan tujuan pembuatan *aplikasi* tersebut, *aplikasi* mempunyai arti yaitu pemecahan masalah yang menggunakan salah satu teknik pemrosesan data *aplikasi* yang biasanya berpacu pada sebuah komputasi yang diinginkan atau diharapkan maupun pemrosesan data yang diharapkan (Nurmaesah dkk, 2017).

Berdasarkan beberapa pengertian diatas dapat disimpulkan bahwa *aplikasi* merupakan suatu perangkat lunak atau program komputer yang beroperasi pada sistem tertentu yang diciptakan dan dikembangkan untuk melakukan perintah tertentu.

1. **Inventarisasi**

Inventarisasi merupakan pencatatan pendaftaran barang-barang milik kantor yang dipakai dalam melaksanakan tugas (Taupik, 2017). Inventarisasi adalah kegiatan untuk mencatat dan menyusun barang-barang atau bahan (aset) yang ada secara benar menurut ketentuan yang berlaku (Singgih dkk, 2017). Inventarisasi dilakukan dalam rangka penyempurnaan pengurusan dan pengawasan yang efektif terhadap aset milik negara atau swasta dan juga memberikan masukan yang sangat berharga bagi efektifitas pengelolaan sarana dan prasarana. Berdasarkan beberapa pengertian tersebut dapat disimpulkan bahwa inventarisasi adalah pencatatan atau pendaftaran barang-barang milik kantor, sekolah, rumah tangga, dan lain sebagainya yang dipakai dalam melaksanakan tugas.

1. **Aset Desa**

Aset Desa adalah barang milik desa yang berasal dari kekayaan asli milik desa, dibeli atau diperoleh atas beban Anggaran Pendapatan dan Belanja Desa (APB Desa) atau perolehan hak lainnya yang sah (Permendagri No.1, 2016).

Menurut Permendagri No. 1 tahun 2016 pasal 2 ayat (1) jenis aset desa terdiri atas:

1. Kekayaan asli desa;
2. Kekayaan milik desa yang dibeli atau diperoleh atas APB desa;
3. Kekayaan desa yang diperoleh dari hibah dan sumbangan atau yang sejenis;
4. Kekayaan desa yang diperoleh sebagai pelaksanaan dari perjanjian/kontrak dan/atau diperoleh berdasarkan ketentuan peraturan undang-undang;
5. Hasil kerja sama desa; dan
6. Kekayaan desa yang berasal dari perolehan yang sah.

Yang dimaksud kekayaan asli desa sebagaimana tercantum dalam Permendagri Nomor 1 tahun 2016 pasal 2 ayat (1) huruf a dijelaskan dalam Permendagri Nomor 1 tahun 2016 pasal 2 ayat (2) terdiri atas:

1. Tanah kas desa;
2. Pasar desa;
3. Pasar hewan;
4. Tambatan perahu;
5. Bangunan desa;
6. Pelelangan ikan yang dikelola oleh desa;
7. Pelelangan hasil pertanian;
8. Hutan milik desa;
9. Mata air milik desa;
10. Pemandian umum; dan
11. Lain-lain kekayaan asli desa.
12. **Desa**

Desa adalah desa dan desa adat atau yang disebut dengan nama lain, selanjutnya disebut Desa, adalah kesatuan masyarakat hukum yang memiliki batas wilayah yang berwenang untuk mengatur dan mengurus urusan pemerintahan, kepentingan masyarakat setempat berdasarkan prakarsa masyarakat, hak asal-usul dan/atau hak tradisional yang di akui dan dihormati dalam sistem peerintahan Negara Kesatuan Republik Indonesia (Undang-Undang RI No 6, 2014). Menurut pasal 1 huruf (a) UU Nomor 5 Tahun 1979 tentang Pemerintahan Desa, desa adalah kesatuan masyarakat hukum yang berhak mengatur dan mengurus rumah tangganya sendiri yang mempunyai organisasi pemerintahan terendah langsung di bawah Camat.

1. ***Website***

*Website* atau disingkat *web, dapat* diartikan sekumpulan halaman yang terdiri dari beberapa laman yang berisi informasi dalam bentuk data digital baik berupa *text,* gambar, video, audio dan animasi lainnya yang disediakan melalui jalur koneksi internet (Abdulloh, 2016). Menurut Prasetyo (2015) *website* merupakan kumpulan berbagai halaman media informasi dalam suatu *domain* yang dapat diakses oleh siapapun menggunakan jaringan internet. Dari beberapa pengertian tersebut dapat disimpulkan bahwa *website* adalah suatu dokumen berupa kumpulan halaman *web* yang saling terhubung dan isinya terdiri dari berbagai informasi berbentuk teks, suara, gambar, video, dan lainnya, dimana semua data tersebut disimpan pada *server* *hosting*. Jenis-jenis *website* berdasarnya sifatnya yaitu:

1. *Web* Statis

*Web* Statis merupakan jenis *web* dimana informasi yang ada di dalamnya tidak dapat di *update* melalui *aplikasi* *website* tersebut melainkan harus merubah *script* yang ada didalamnya.

1. *Web* Dinamis

*Web* Dinamis merupakan jenis *website* dimana informasi yang ada di dalamnya dapat di *update* melalui *aplikasi* *website* tersebut dan kita dapat membuat *aplikasi* *Content Management System* atau CMS yang bisa kita gunakan untuk meng*update* informasi terbaru.

1. ***Framework***

*Framework* adalah kerangka kerja. *Framework* juga dapat diartikan sebagai kumpulan *script* (terutama *class* dan *functio*n) yang dapat membantu *developer* atau *programer*, seperti koneksi ke *database*, pemanggilan variabel*, file* dan lain-lain sehingga pekerjaan *develope*r lebih fokus dan lebih cepat dalam pembangunan *aplikasi* (Yudhananto & Prasetyo,2018).

*Framework* atau kerangka kerja pengembangan *aplikasi* adalah suatu standar yang harus diikuti untuk melakukan pengembangan *aplikasi* oleh pemrogram. Standar ini mengatur banyak hal, mulai dari nama *file*, direktori, dan cara memrogramnya. *Framework* memberikan kerangka program, kumpulan *library* dan fungsi yang bisa langsung digunakan, serta aturan untuk menggunakannya (Sidik, 2018).

*Framework secara* sederhana dapat di artikan sebagai kumpulan dari fungsi atau prosedur dan *class* untuk tujuan tertentu yang sudah siap untuk digunakan sehingga mempermudah dan mempercepat *programer* dalam pembuatan program tanpa harus membuat fungsi atau *class* dari awal (Ardhana dan Budiyanto, 2018).

Dari penjelasan di atas dapat disumpulkan bahwa *framework* adalah sebuah kerangka kerja sebagai sekumpulan fungsi dan *class* yang sudah siap digunakan untuk mempermudah dan mempercepat kerja dari *developer* atau *programer*.

1. ***CodeIgniter***

*CodeIgniter* (CI) adalah *framework* pengembangan *aplikasi* (*Application Development Framework*) dengan menggunakan PHP, suatu kerangka untuk berkerja atau membuat program dengan menggunakan PHP yang lebih sistematis. Pemrogram tidak perlu membuat program dari awal (*from scratch*), karena CI menyediakan sekumpulan *library* dan fungsi yang banyak, yang diperlukan untuk menyelesaikan pekerjaan yang umum, dengan menggunakan antarmuka dan struktur logika yang sederhana untuk mengakses *librarynya*. Pemrogram dapat memfokuskan diri pada kode yang harus dibuat untuk menyelesaikan suatu pekerjaan (Sidik, 2018)

*CodeIgniter* adalah sebuah *aplikasi* *open source* yang berupa kerangka kerja atau *framework* untuk membangun *website* menggunakan bahasa pemrograman PHP. Tujuannya memungkinkan pengembangan proyek yang lebih cepat dari pada penulisan kode dasar atau kode terstruktur, dengan menyediakan banyak *library* yang biasanya digunakan dalam pengerjaan. Antarmuka yang sederhana dan terstruktur logika untuk mengakses *library* ini membuat *CodeIgniter* mudah digunakan dan dipelajari. *CodeIgniter* ditulis atau dibuat oleh Ellis Lab dan dirilis pertama kali pada 28 Februari 2006 (Sulistiono,2018)

*CodeIgniter* disebut sebagai *framework* pengembangan *aplikasi* (*Application Development Framework*) dengan menggunakan PHP, dengan kerangka kerja sehingga menjadi sistematis . Dalam hal ini *programer* tidak perlu membuat *website* dari awal (*from scratch*), karena *CodeIgniter* telah menyediakan *library* yang dapat dimanfaatkan untuk menyelesaikan pekerjaan yang umum, dengan menggunakan antarmuka dan struktut yang lebih sederhana untuk mengakses *librarynya* (Ardhana dan Budiyanto, 2018).

Saat ini, *CodeIgniter* dikembangkan oleh komunitas dan disebarluaskan ke seluruh dunia dengan lisensi bebas. *CodeIgniter* tergolong *framework* dengan ukuran kecil dan cukup mudah dipahami dan dikuasai.

Menurut Ardhana dan Budiyanto (2018) kelebihan dalam membuat *website* dengan menggunakan *framework CodeIgniter,* diantaranya yaitu:

1. *Open Source*

*CodeIgniter* merupakan *framework* yang gratis untuk digunakan dan dikembangkan secara legal. *CodeIgniter* berlisensi *Apache* atau BSD-*style* *open source* sehingga dapat dikembangkan dan digunakan sesuai dengan kebutuhan.

1. *Multiplatform*

*CodeIgniter* sangat ringan ketika dijalankan pada berbagai *platform*, pada bagian sistem utama *CodeIgniter* hanya memerlukan kapasitas yang sedikit untuk *server* dalam membentuk *class* *library*.

1. Efisiensi Waktu

Dengan adanya struktur dan *library* yang telah disediakan oleh *framework* maka tidak lagi memikirkan hal-hal tersebut, hal ini dapat membuat *programer* fokus pada proses pengembangan sistem yang akan dibangun.

1. Menggunakan metode MVC

Metode MVC adalah metode dengan cara Model, *View*, *Controller*. Dengan model *framework* ini maka dapat mempermudah dalam membedakan antara tampilan dan program.

1. *User Friendly*

*CodeIgniter* sangat mudah digunakan oleh berbagai kalangan *user* atau *programer*, hal ini dikarenakan bentuk dan tampilan yang sudah dirancang secara terstruktur.

1. URL yang *User* *Friendly*

Ketika URL dipanggil maka akan langsung membentuk sebuah alamat http://namasitus.com/blog/contoh/alamat-blog hal ini dikarenakan *CodeIgniter* telah dirancang dengan rapi.

1. *Framework* yang lengkap

*CodeIgniter* telah dikemas secara lengkap karena di dalamnya terdapat kumpulan *class* yang ada di dalam *library* yang telah tersedia sehingga sangat menunjang dalam pembuatan *website*, *database*, validasi *form* data, pembuatan *session*, pengguna XML-RPC, dan lain-lain.

1. *User Guide*

Di dalam *CodeIgniter* telah dilengkapi dengan pendokumentasian yang cukup baik dan lengkap, sehingga dapat membantu para *programer* dalam mempelajari lebih jauh tentang *CodeIgniter*.

1. *Reuse of code*

Dengan menggunakan *framework* maka program yang dibuat akan memiliki struktur yang baku. Sehingga dapat digunakan kembali pada proyek-proyek lainnya.

1. Komunitas

Komunitas saat ini komunitas *CodeIgniter* sudah mulai berkembang sehingga dapat dijadikan acuan dalam mencari ilmu dalam memecahkan suatu masalah yang dihadapi, selain itu dapat untuk meningkatkan pengetahuan akan kemampuan pemrograman dengan menggunakan *framework CodeIgniter.*

1. Kumpulan *Best Pracite*

Sebuah *framework* merupakan kumpulan *best* *practice* yang sudah diuji. Jadi dapat digunakan untuk meningkatkan kualitas program atau sistem yang dibangun.

1. Berjalan di PHP versi 5 dan 7

Sekarang ini PHP sudah mencapai versi ke 7, meskipun begitu masih banyak orang yang tetap menggunakan PHP versi 5, oleh sebab itu *CodeIgniter* dikembangkan agar tetap kompatibel dengan PHP versi 5 dan dapat dijalankan pada PHP versi 7.

1. **UML (*Unified Modeling Language*)**
2. Pengertian UML

UML (Unified Modeling Language) merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks – teks pendukung. Banyak orang yang telah membuat bahasa pemodelan pembangunan perangkat lunak sesuai dengan teknologi pemrograman yang berkembang pada saat itu, misalnya yang sempat berkembang dan digunakan oleh banyak pihak adalah Data Flow Diagram (DFD) untuk memodelkan perangkat lunak yang menggunakan pemrograman prosedural atau struktural, kemudian juga ada State Transition Diagram (STD) yang digunakan untuk memodelkan sistem real time (waktu nyata). Pada perkembangan teknik pemrograman berorientasi objek, munculah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek, yaitu Unifed Modeling Language (UML) (Rosa dan Salahudin, 2018).

1. *Class Diagram*

Diagram kelas atau class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi. Atribut merupakan variabel*-*variabel yang dimiliki oleh suatu kelas. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas. Diagram kelas dibuat agar pembuat program atau programermembuat kelas – kelas sesuai rancangan di dalam diagram kelas agar antara dokumentasi perancangan dan perangkat lunak sinkron. Banyak berbagai kasus, perancangan kelas yang dibuat tidak sesuai dengan kelas-kelas yang dibuat pada perangkat lunak, sehingga tidaklah ada gunanya lagi sebuah perancangan karena apa yang dirancang dan hasil jadinya tidak sesuai (Rosa dan Salahudin, 2018).

Berikut adalah simbol-simbol yang ada pada diagram kelas:

Tabel 2.1 Simbol-Simbol pada *Class Diagram*

|  |  |
| --- | --- |
| Simbol | Deskripsi |
| Kelas | Kelas pada struktur sistem. |
| Antarmuka/*Interface*    **nama\_*interface*** | Sama dengan konsep *interface* dalam pemrograman berorientasi objek. |
| Asosiasi/*Association* | Relasi antar kelas dengan makna umum, asosisasi biasanya juga disertai dengan *multiplicity.* |
| Asosiasi Berarah/*Directed Association* | Relasi antar kelas dengan makna kelasa yang satu digunakan oleh kelas yang lain, asosisasi biasanya juga disertai dengan *multiplicity.* |

Sumber : (Rosa dan Salahudin, 2018).

Tabel 2.1 Simbol-Simbol pada *Class Diagram* (Lanjutan)

|  |  |
| --- | --- |
| Simbol | Deskripi |
| Generalisasi | Relasi antar kelas dengan makna generalisasi-spesialis (umum khusus). |
| Kebergantungan/*Dependency* | Relasi antar kelas dengan makna kebergantungan. |
| Agregasi/*Aggregation* | Relasi antar kelas dengan makna semua-bagian (*whole part*). |

Sumber : (Rosa dan Salahudin, 2018).

1. *Use Case Diagram*

*Use case* atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu (Rosa dan Salahudin, 2018).

Berikut simbol-simbol yang ada pada *diagram use case*:

Tabel 2.2 Simbol-Simbol pada *Use Case Diagram*

|  |  |
| --- | --- |
| Simbol | Deskripsi |
| *Use case* | Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case.* |

Sumber : (Rosa dan Salahudin, 2018).

Tabel 2.2 Simbol-Simbol pada *Use Case Diagram* (Lanjutan)

|  |  |
| --- | --- |
| Simbol | Deskripsi |
| Aktor    **nama aktor** | Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang biasanya dinyatakan menggunakan kata benda di awal frase nama aktor. |
| Asosiasi/*Association* | Komunikasi antara aktor dan *use* *case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor. |
| Ekstensi/*Extend* | Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan itu; mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek; biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan, misal    Arah panah mengarah pada *use case* yang ditambahkan; biasanya *use case* yang menjadi *extend*-nya merupakan jenis yang sama dengan *use case* yang menjadi induknya. |

Sumber : (Rosa dan Salahudin, 2018).

Tabel 2.2 Simbol-Simbol pada *Use Case Diagram* (Lanjutan)

|  |  |
| --- | --- |
| Simbol | Deskripsi |
| Generalisasi/*Generalization* | Hubungan generalisasi dan spesialiasi (umum – khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya, misalnya :    Arah panah mengarah pada use caseyang menjadi generalisasinya (umum). |
| Menggunakan/*Include*/*Uses* | Relasi *use case* tambahan ke sebuah *use case* di mana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini ada dua sudut pandang yang cukup besar mengenai *include* di *use* *case*:   1. *Include* berarti *use case* yang ditambahkan akan selalu dipanggil saat *use cae* tambahan dijalankan misal pada kasus berikut : |

Sumber : (Rosa dan Salahudin, 2018)

Tabel 2.2 Simbol-Simbol pada *Use Case Diagram* (Lanjutan)

|  |  |
| --- | --- |
| Simbol | Deskripsi |
|  | b. *Include* berarti use case yang tambahan akan selalu melakukan pengecekan apakah use case yang ditambahkan telah dijalankan sebelum use case tambahan dijalankan, misal pada kasus berikut :    Kedua interpretesi diatas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretesi yang dibutuhkan. |

Sumber : (Rosa dan Salahudin, 2018)

1. *Activity Diagram*

Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem (Rosa dan Salahudin, 2018).

Berikut simbol-simbol yang ada pada diagram aktivitas:

Tabel 2.3 Simbol-Simbol pada *Activity Diagram*

|  |  |
| --- | --- |
| Simbol | Deskripsi |
| Status awal | Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal |
| Aktivitas | Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja. |
| Percabangan/*Decisison* | Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu. |
| Penggabungan/*Join* | Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu. |
| Status akhir | Status akhir dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir. |
| *Swimlane* | Memisahkan organisasi bisnis yang bertanggung jawab terhadap yang terjadi. |

Sumber : (Rosa dan Salahudin, 2018).

1. *Sequence Diagram*

Diargam sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambar diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat skenario yang ada pada *use case*. Banyaknya diagram sekuen yang harus digambar adalah minimal sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak *use case* yang didefinisikan maka diagram sekuen yang harus dibuat juga semakin banyak (Rosa dan Salahudin, 2018).

Berikut adalah simbol-simbol yang ada pada diagram sekuen:

Tabel 2.4 Simbol-Simbol pada *Squence Diagram*

|  |  |
| --- | --- |
| Simbol | Deskripsi |
| Garis hidup/*lifeline* | Menyatakan kehidupan suatu objek. |
| Objek | Menyatakan objek yang berinteraksi pesan. |
| Waktu aktif | Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya, misalnya    Maka cek *statuslogin*() dan *open*() dilakukan di dalam metode *login*(). Aktor tidak memiliki waktu aktif. |
| Pesan tipe *create* | Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat. |

Sumber : (Rosa dan Salahudin, 2018).

Tabel 2.4 Simbol-Simbol pada *Squence Diagram* (Lanjutan)

|  |  |
| --- | --- |
| Simbol | Deskripsi |
| Pesan tipe *call* | Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau diterima sendiri,    Arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi/metode maka operasi/metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi. |

Sumber : (Rosa dan Salahudin, 2018).

### *Programming*

#### HTML

HTML (HyperText Markup Language) adalah sebuah bahasa markah yang digunakan untuk membuat sebuah halaman web, menampilkan berbagai informasi seperti gambar, teks, video dan suara pada penjelajah web internet, yang ditulis dalam berkas format ASCII agar dapat menghasilkan tampilan wujud yang terintregasi (Sulistiono, 2018).

HTML atau sering disebut dengan Hyper-Text Murkup Language, awalnya dulu pernah disebut sebagai pemrograman. Dari kepangjangannya, bisa disimpulkan bahwa HTML adalah sebuah penkita atau mark (Prasetyo dan Yudhanto, 2018).

#### *Javascript*

*Javascript* adalah bahasa pemrograman tingkat tinggi untuk membuat *website* dinamis, dan *Jquery* adalah *library* atau pustaka dari *javascript* yang dirancang untuk memudahkan penerapan *client side scripting* dan menyajikan sebuah paradigma baru pada penanganan *event* pada *Javascript* (Sulistiono, 2018).

*Javascript* atau sering disingkat dengan JS. JS digunakan untuk membuat *website* kita lebih interaktif kepada *user* yang menggunakan *website* tersebut. Misal, kita ingin mengubah suatu nilai pada HTML dengan meng-klik satu tombol, kita bisa pakai *Javascript* ini (Prasetyo dan Yudhanto, 2018).

#### PHP *(Hypertext Preprocessor)*

#### PHP (Hypertext Preprocessor) adalah bahasa pemrograman yang digunakan untuk membuat *website* atau situs dinamis dan menangani rangkaian bahasa pemograman antara client side scripting dan server sidescripting (Sulistiono, 2018).

#### PHP atau Hypertext Preprocessor adalah bahasa pemrograman 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 webyang bersifat dinamis (Prasetyo dan Yudhanto, 2018).

#### *MySQL*

*MySQL* adalah salah satu jenis *database server* yang sangat terkenal. Kepopulerannya disebabkan *MySQL* menggunakan SQL sebagai bahasa dasar untuk mengakses *database*nya. *MySQL* termasuk jenis RDBMS (*Relational Database Management System*). Pada *MySQL*, sebuah *database* mengandung satu atau sejumlah tabel. Tabel terdiri atas sejumlah baris dan setiap baris mengandung satu atau beberapa kolom (Sutopo, Cahyadi, & Arifin, 2016)

*MySQL* adalah sistem manajemen *database* SQL yang sifatnya *open source* (terbuka) dan paling banyak digunakan saat ini. Sistem *database MySQL* mampu mendukung beberapa *fitur* seperti *multithreaded, multi-user,* dan SQL *database management system* (DBMS) (Mundzir,2018). Kelebihan *MySQL* sebagai berikut :

1. *Portabilitas*

Dengan kelebihan ini, *MySQL* dapat berjalan stabil pada berbagai sistem operasi seperti *Windows*, *Linus*, *FreeBSD*, *Mac* OS X *Server*, *Solaris*, *Amiga*, dan masih banyak lagi.

1. *Open Source*

Keunggulan ini biasa dikenal sebagai perangkat lunak sumber terbuka yang mengakibatkan *MySQL* didistribusikan dan digunakan secara gratis.

1. *Multi*-*User*

Keunggulan ini membuat *MySQL* dapat digunakan oleh beberapa pengguna dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.

1. *Perfromance Tunning*

Hal ini berkaitan dengan kecepatan akses, yang mana *MySQL* memiliki kecepatan yang menakjubkan dalam menanganni *query* sederhana dan mampu memproses lebih banyak SQLper satuan waktu.

1. Ragam tipe data

Ragam tipe data dari *MySQL* yang kaya, seperti *signed/unsigndeinteger, float, double, char, text, date, timestamp,* dan lain – lain.

1. Perintah dan fungsi

*MySQL* memiliki operator dan fungsi secara penuh yang mendukung perintah *select* dan *where* dalam perintah (*query*).

1. Keamanan

Lapisan keamanan dari *MySQL* dengan menggunakan *password* yang terenkripsi.

1. Sakalabilitas dan pembatasan

*MySQL* mampu menangani bisnis data dalam skala besar dengan jumlah *record* lebih dari 50 juta dan 60 ribu tabel serta 5 miliar baris dan batas indeks yang dapat ditampung mencapai 32 indeks pada tiap tabelnya.

1. Koneksi

Protokol TCP/IP, *Unix* soket (*UNIX*), atau *named pipes* (NT) merupakan koneksi yang mampu dilakukan oleh *MySQL.*

1. **Perangkat Lunak Yang Digunakan**
2. *XAMPP*

*XAMPP* merupakan singkatan dari X (empat sistem operasi apapun), *XAMPP* merupakan perangkat lunak bebas, yang mendukung banyak sistem operasi, dan merupakan kompilasi dari beberapa program. Seperti *Apache, MySQL*, PHP, dan *Perl* (Novendri dkk, 2019)*.*

*XAMPP* adalah kompilasi program *aplikasi* gratis terfavorit di kalangan developer/programer yang berguna untuk pengembangan website berbasis PHP dan *MySQL*. Dengan satu *aplikasi* ini, kita akan mendapatkan paket software komplit yang bisa dijalankan pada *Windows* ataupun *Linux* sehingga *programer* dapat dengan mudah melakukan simulasi pada komputer *local* sebelum diunggah ke internet. Software *XAMPP* dibuat dan dikembangkan oleh *Apache* *Friends* (Prasetyo dan Yudhanto, 2018).

1. PHP *Myadmin*

*PhpMyAdmin* merupakan *aplikasi* berbasis *web* yang digunakan untuk membuat *database* *MySQL* sebagai tempat untuk menyimpan data-data *website* (Abdulloh, 2016).

*PhpMyAdmin* adalah *aplikasi* *web* untuk mengelola *database* *MySQL* dan *database* MariaDB dengan mudah melalui antarmuka (*interface*) grafis. *Aplikasi* *web* ini ditulis menggunakan bahasa pemrograman PHP. Sebagaimana *aplikasi*-*aplikasi* lain untuk lingkungan *web* (*aplikasi* yang dibuka atau dijalankan menggunakan *browser*), *phpMyAdmin* juga mengandung unsur HTML/XHTML, CSS, dan juga kode JavaScript (Prasetyo dan Yudhanto, 2018).

1. *Browser*

*Browser* adalah suatu *aplikasi* atau program yang dijalankan pada perangkat komputer untuk melihat konten yang ada pada media *World* *Wide* *Web* (WWW) dengan memanfaatkan jaringan internet. Teknologi *browser* yang berkembang saat ini tidak hanya dapat menampilkan halaman yang berisi *text* atau tulisan saja, *browser*-*browser* populer sekarang dapat menampilkan gambar, musik, suara, video, *file* pdf dan data lainnya (Marisa dkk, 2017).

*Browser* digunakan untuk menampilkan hasil *website* yang telah dibuat. *Web* *browser* yang sering digunakan, di antaranya *Mozila* *Firefox*, *Google* *Chrome*, *Intrenet* *Explorer*, *Opera*, dan Safari (Abdulloh, 2016).

1. **Penelitian Sebelumnya**

Dalam melakukan penelitian ini, peneliti mengambil referensi dari penelitian sebelumnya dengan tema yang sama yang pernah dilakukan oleh peneliti lain. Penelitian sebelumnya yang dijadikan sebagai referensi dalam penelitian ini diantaranya:

1. Penelitian yang dilakukan oleh Maulana Malik dan Ardi Mardiana (2018), yang berjudul “Rancang Bangun Sistem Informasi Pengelolaan Aset Desa Rajagaluh Kidul”. Tujuan pada penelitian tersebut adalah membantu Pemerintahan Desa Rajagaluh Kidul dalam pengelolaan data aset sehingga mempermudah dan mempercepat proses pengelolaan data aset. Pada penelitian tersebut *aplikasi* yang dibuat masih berbasis *web offline*.
2. Penelitian yang dilakukan oleh Titus Aditya Kinaswara, Nasrul Rofi’ah Hidayati, Fatim Nugrahanti (2019), yang berjudul “Rancang Bangun *Aplikasi* Inventaris Berbasis *Website* pada Kelurahan Bantengan”. Tujuan penelitian tersebut yaitu untuk memberikan kemudahan dalam mengelola data inventaris di Kelurahan Bantengan dan bertujuan untuk terciptanya sebuah *aplikasi* inventarisasi barang berbasis *website* pada kelurahan bantengan. Pada penelitian tersebut *aplikasi* yang dibuat hanya untuk inventarisasi barang, tidak untuk semua aset desa yang dimiliki.
3. Penelitian yang dilakukan oleh Adi Sakti Almajid (2018), yang berjudul “Pengembangan Sistem Inventori Aset Berbasis *Website* Pada Desa Sokokulon”. Tujuan penelitian tersebut yaitu untuk memudahkan proses inventarisasi aset secara terpadu. *Aplikasi* yang dibuat pada penelitian tersebut hanya untuk mengelola data aset desa yang dapat dioperasikan oleh staf desa terkait, belum dapat mencetak laporan data aset desa tersebut.

Berikut merupakan hasil perbandingan penelitian sebelumnya ditunjukan pada Tabel 2.1 dibawah ini:

Tabel 2.1 Perbandingan Penelitian Sebelumnya

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Penulis | Masalah | Persamaan | Perbedaan | |
| Hasil Penelitian Terdahulu | Rencana Penelitian dan Kelebihan |
| 1 | Maulana Malik dan Ardi Mardiana (2018) | Pengelolaan aset yang masih dilakukan dengan cara manual. | Membuat *aplikasi* inventarisasi aset. | Menghasilkan *aplikasi* pengelolaan data aset namun masih *offline*. | Membuat *aplikasi* pengelolaan aset yang bisa di akses secara *online*. |
| 2 | Titus Aditya Kinaswara, Nasrul Rofi’ah Hidayati, dan Fatim Nugrahanti (2019) | Penurunan kinerja pegawai karena kesalahan pembuatan laporan. | Membuat *aplikasi* inventarisasi aset. | Menghasilkan *aplikasi* pengelolaan data barang inventaris yang bisa di cetak secara periodik. | Membuat *aplikasi* inventarisasi yang tidak hanya mendata barang tetapi aset lainnya dan membuat tampilan website menjadi lebih responsive. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Penulis | Masalah | Persamaan | Perbedaan | |
| Hasil Penelitian Terdahulu | Rencana Penelitian dan Kelebihan |
| 3. | Adi Sakti Almajid (2018) | Proses pencatatan masih menggunakan cara yang konvensional yaitu dengan menggunakan buku sehingga menyebabkan beberapa permasalahan seperti redudansi data pencatatan. | Membuat *aplikasi* inventarisasi aset. | Menghasilkan *aplikasi* pengelolaan data aset namun tidak adanya *fitur* peminjaman. | Membuat *aplikasi* pengelolaan aset yang bisa menyediakan *fitur* peminjaman. |