# BAB II

# TINJAUAN PUSTAKA DAN LANDASAN TEORI

## 2.1 Tinjauan Pustaka

Dalam penelitian ini penulis mengacu pada beberapa penelitian sebelumnya.

Muhammad Nur Hamid (2019) dalam penelitiannya melakukan analisis perbandingan antara *framework codeigniter* dan *framework laravel* pada aplikasi inventaris HMJ TI STMIK AKAKOM YOGYAKARTA agar mendapatkan hasil perbandingan antara *framework codeigniter* dan *framework laravel* dari segi performa, cara akses *databas*e, dan implementasi fitur *AJAX*.

Nugroho (2007) dari Universitas Gadjah Mada pada penelitiannya yang melakukan analisis berjudul (implementasi AJAX menggunakan Atlas Framework untuk pembuatan Web Layanan Informasi Pariwisata Jogja) membahas tentang membangun aplikasi web pada pariwisata Jogja dengan AJAX, dengan menggunakan AJAX memiliki loading lebih singkat karna melewatkan data lebih sedikit jika dibandingkan dengan web tradional pada umumnya, akan tetapi AJAX memiliki kekurangan yaitu kode sumbernya yang susah ditulis. Beberapa framework hadir untuk mengatasi masalah ini, salah satunya Atlas Framework.

Sasmito (2017) dari Institut Teknologi Nasional Malang yang berjudul (Implementasi AJAX pada Peta Wisata Esbatu Sistem Informasi Jejaring Wisata Kota Batu) membahas tentang pengembangan peta wisata menggunakan AJAX

dengan menggunakan *waterfall* model dengan pengujian BlackBox testing serta menerapkan *DeLone and McLean Model of Information System Succes* untuk mengukur validasi peta wisata.

Fitriya (2015) yang berjudul (Sistem Iventori Barang KP.Market Mataram mengggunakan AJAX) membahas tentang perancangan sistem iventori barang sehingga transaksi pembelian dan penjualan bisa dikelola melalui sistem, informasi yang diperoleh juga lebih cepat sehingga keputusan dapat diambil lebih cepat dan tepat.

Perbandingan yang membedakan penelitian ini dengan penilitian sebelumnya adalah adanya pendaftaran yang mengharuskan seseorang yang ingin menjadi anggota harus mendaftar terlebih dahulu di koprasi, adanya dua *role* untuk *login* .

**Tabel 2.1 Tinjauan Pustaka**

| **Penulis** | **Obyek** | **Metode/Teknologi** | **Bahasa Pemrograman** | **Interface** |
|---|---|---|---|---|
| Muhammad Nur Hamid (2019) | Inventaris di HMJ TI STMIK AKAKOM Yogyakarta | Perbandingan performa, cara akses database, dan implemenetasi fitur AJAX | PHP | Text |
| Nugroho (2007) | Layanan Informasi Pariwisata Yogyakarta | *AJAX*<br>*Atlas Framework* | PHP | Web |
| Sasmito (2017) | Peta Wisata Es Batu | AJAX | PHP | Web |
| Fitria (2015) | Iventory Barang pada KP. Market Mataram | *AJAX* | PHP | Web |
| LL. Mahadir Muhamad (2020) | Koprasi Simpan Pinjam Desa Pengkelak Mas | *AJAX* | PHP | Web |

## 2.2 Dasar Teori

### 2.2.1 Java Script

Javascript dikembangkan oleh Netscape Communications yang bekerja sama dengan Sun Microsystem. Sebenarnya javasript dikembangkan dari bahasa livescript yang khusus dirancang untuk netscape navigator. Oleh

karena itu dulunya javascript hanya dapat dijalankan pada *browser* netscape navigator, namun dalam perkembangannya Internet Explorer yang dirilis oleh Microsoft juga dapat mendukung javascript. Javascript memang tidak secanggih java ( karena hanya merupakan *script* ) tetapi untuk pemrogramannya tidak diperlukan pemrogram profesional. Javascript dijalankan di *client* ( *client side programing* ) sehingga dapat mengurangi beban kerja dari *serve*r.

Dengan javascript dapat dibuat halaman *web* yang interaktif dan juga cerdas. Sebagai contoh javascript dapat digunakan untuk mengecek sah tidaknya masukan pengguna sebelum masukan dikirim ke *server*, javascript juga dapat melakukan operasi aritmatik ( seperti penjumlahan, pengurangan, dan perkalian ), dan bisa juga menampilkan animasi sederhana.

Javascript bukan bahasa berorientasi *object*, melainkan bahasa berbasis *object*. Bahasa berorientasi *object* harus mendukung tiga konsep dasar, yaitu pengkapsulan (*encapsulation*), pewarisan (*inheritance*), dan polimorfisme (*polymorphism*). Javascript hanya mendukung pengkapsulan, dan itupun tidak 100%.

Program javascript dituliskan pada file HTML ( .html atau .htm )dengan menggunakan tag kontainer <SCRIPT>. Artinya javascript tidak perlu dituliskan pada file terpisah. Tag kontainer adalah tag yang diawali dengan <NAMA_TAG> dan diakhiri dengan </NAMA_TAG>. Beberapa contoh tag kontainer adalah <HTML></HTML>, <HEAD></HEAD>,

<BODY></BODY>, dan sebagainya. Tag kontainer <SCRIPT> mempunyai dua atribut tetapi yang harus diisi hanya satu atribut, yaitu *Language* yang diisi dengan "Javascript". Hal ini memberitahu browser bahwa script yang ditulis adalah javascript [Antony Pranata, 1997]. Contoh :

<SCRIPT LANGUAGE="Javascript">
// program javascript ditulis di bagianini
</SCRIPT>

### 2.2.2 XML

XML kependekan dari *eXtensible Markup Language*, yang dikembangkan mulai tahun 1996 dan mendapatkan pengakuan dari W3C pada bulan Februari 1998. Teknologi yang digunakan pada XML merupakan turunan dari SGML yang telah dikembangkan pada awal tahun 80-an dan telah banyak digunakan pada dokumentasi teknis proyek-proyek berskala besar. Ketika HTML dikembangkan pada tahun 1990, para pengembang XML mengadopsi bagian paling penting pada SGML dan dengan berpedoman pada pengembangan HTML menghasilkan *markup language* yang tidak kalah hebatnya dengan SGML.

Seperti halnya HTML, XML juga menggunakan *elemen* yang ditandai dengan tag pembuka (diawali dengan '<' dan diakhiri dengan '>'), tag penutup(diawali dengan '</ 'diakhiri '>') dan atribut elemen (parameter yang

dinyatakan dalam tag pembuka misal <form name=”isidata”>).

Perbedaannya, HTML mendefinisikan dari awal tag dan atribut yang dipakai di dalamnya, sedangkan pada XML dapat menggunakan tag dan atribut sesuai keinginan dan kebutuhan yang membuatnya. Pada XML, penyimpanan data baik dalam atribut maupun sebagai isi elemen yang diletakkan diantara *tag* pembuka dan *tag* penutup.

HTML digunakan untuk menampilkan informasi dan berfokus pada bagaimana informasi ditampilkan, sedangkan XML mendeskripsikan susunan informasi dan berfokus pada informasi itu sendiri. XML terutama dibutuhkan untuk menyusun dan menyajikan informasi dengan format yang tidak mengandung format standar seperti *heading*, *paragraph*, *table* dan lain sebagainya. XML lebih bersifat fleksible dibandingkan dengan HTML dalam hal kemampuannya menyimpan informasi dan data. Kelebihan lain yang dimiliki XML adalah informasi dapat dipertukarkan dari satu sistem ke sistem lain yang berbeda *platform*. Misalnya dari Windows ke Unix, atau dari PC ke Macintosh bahkan dari internet ke *handphone* dengan teknologi WAP [Ady Wicaksono, 2004].

### 2.2.3 AJAX Sebagai Kombinasi Beberapa Teknologi

AJAX akronim dari *Asynchronous JavaScript and XML*. AJAX bukanlah suatu teknologi yang berdiri sendiri, tetapi suatu kombinasi yang mengacu pada penggunaan beberapa teknologi secara bersama – sama. Jadi AJAX dapat

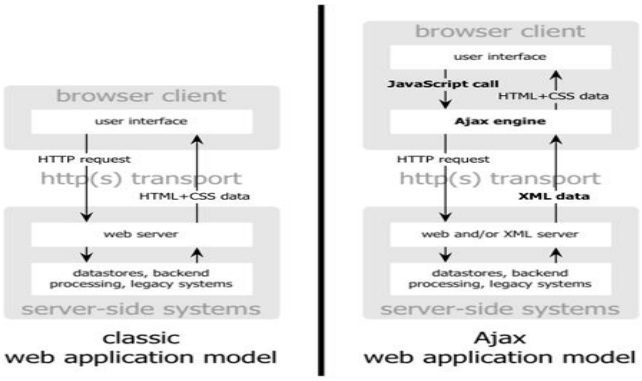
diartikan sebagai "*Asynchronous JavaScript + CSS + DOM + XMLHttpRequest*" Sebagai contoh dapat dilihat pada Gmail, jika diperhatikan jumlah *mail* dalam *inbox*. Jika ada mail baru masuk angka tersebut akan berubah tanpa me*refresh* halaman tersebut. Gmail melakukan itu dengan memakai teknologi *XMLHttpRequest*. *XMLHttpRequest* Adalah komponen teknis yang memungkinkan komunikasi *asynchronous* antara *client* dengan *server*. *XMLHttpRequest* ini terdapat pada *feature* dari *browser - browser* yang relatif baru untuk melakukan *request HTTP*, *feature* ini dapat diakses dengan *JavaScript*. Jadi halaman Gmail tersebut akan me*request* ke *server* jumlah *inbox* yang kita miliki beberapa waktu sekali dan meng*update* nilainya di halaman *web*. Jadi akhirnya halaman itu tidak di *request* ulang, tapi hanya *request* data yang terjadi di belakang layar. Pada model aplikasi *web* klasik proses kerjanya adalah : *client* mengirimkan *HTTP request* untuk meminta respons dari *server*, *server* melakukan beberapa proses seperti : menerima data, mengolahnya di database, dan *server web* merespon dengan pengiriman suatu halaman *web* yang baru yang berisi informasi yang dibutuhkan oleh *client.* Proses suatu aplikasi *web* dengan model AJAX adalah : ketika aplikasi di*load* yang dilakukan adalah me*load AJAX engine*. *AJAX Engine* dibuat dengan javascript yang fungsinya untuk me*render* tampilan untuk *user* dan dan berkomunikasi dengan *server*. Dengan begitu komunikasi ke *server* bisa terjadi secara *asynchronous* [Sams, 2006].

Dengan demikian *AJAX Engine* mengupdate tampilan *browser* di *client* setiap

beberapa waktu sekali sesuai dengan informasi yang didapat dari *server*, dan *client* tidak perlu melihat halaman kosong ketika melakukan *submit* ke *server*. Untuk lebih jelasnya tentang perbandingan proses yang dilakukan oleh aplikasi *web* klasik dan web yang menggunakan AJAX, dapat dilihat pada gambar di bawah ini

*engine*. *AJAX Engine* dibuat dengan javascript yang fungsinya untuk me*render* tampilan untuk *user* dan dan berkomunikasi dengan *server*. Dengan begitu komunikasi ke *server* bisa terjadi secara *asynchronous* [Sams, 2006].

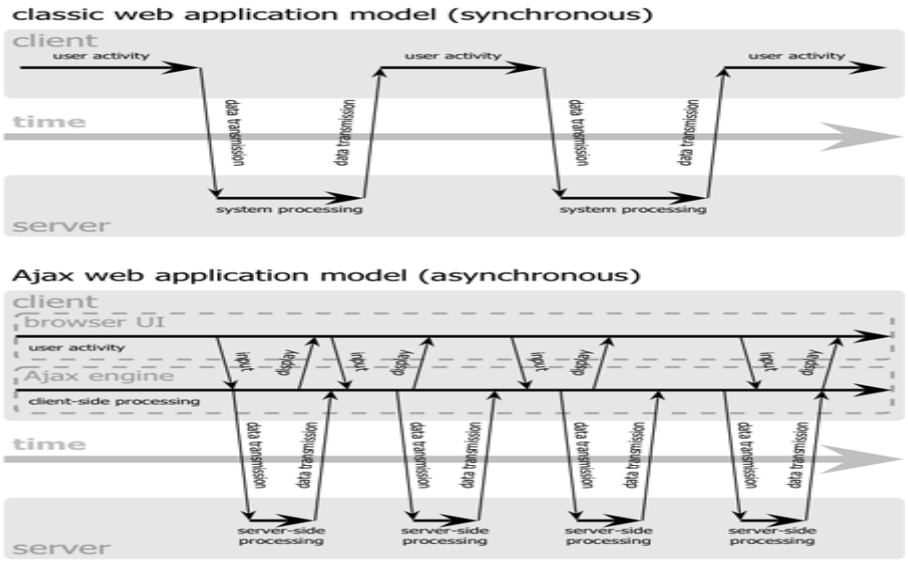
Dengan demikian *AJAX Engine* mengupdate tampilan *browser* di *client* setiap beberapa waktu sekali sesuai dengan informasi yang didapat dari *server*, dan *client* tidak perlu melihat halaman kosong ketika melakukan *submit* ke *server*. Untuk lebih jelasnya tentang perbandingan proses yang dilakukan oleh aplikasi *web* klasik dan web yang menggunakan AJAX, dapat dilihat pada gambar di bawah ini :




**Gambar 2.1 Perbandingan cara kerja Web Klasik dan Ajax**

Perbedaan antara aplikasi *web* berbasis AJAX dengan *web* klasik adalah, AJAX mengeliminasi *start-stop-start-stop* pada interaksi *client* dengan *server*. Dengan menggunakan *AJAX engine*, komunikasi yang terjadi antara *client* dan *server* dapat berlangsung terus menurus secara *asynchronous*, Ini terlihat seperti memasukkan sebuah *layer* pada halaman *web* pengguna tanpa melakukan proses

reload. Untuk lebih jelasnya mengenai komunikasi antara client dan server dengan menggunakan aplikasi web berbasis AJAX dan menggunakan web klasik, dapat dilihat pada gambar di bawah ini :




**Gambar 2.2 Perbandingan Transfer Data**

### 2.2.4 XMLHttpRequest Object

Jika *user* mengklik sebuah *hyperlink* atau *submit* di sebuah halaman HTML maka *browser* akan mengirimkan *HTTP request* ke *server*, kemudian

*server* akan meberikan respons kepada *user* dengan mengirimkan sebuah halaman baru. Untuk aplikasi *web* yang bekerja secara *asynchronously, HTTP request* harus dapat dikirimkan tanpa menampilkan sebuah halaman baru.

Sebagai solusi untuk memecahkan masalah diatas, bisa digunakan *XMLHTTPRequest object. JavaScript object* dapat digunakan untuk membuat koneksi ke *server* dan mengirimkan *HTTP request* tanpa perlu me*load* sebuah halaman baru. Walaupun object *XMLHTTPRequest* diawali dengan kata XML, namun sesungguhnya data yang dapat dikembalikan dari *server* bukan hanya berupa XML, tapi juga yang lain seperti ASCII, text, dan HTML.

*XMLHTTPRequest* disupport oleh semua *browser* modern, Microsoft Internet Explorer 5+, Mozilla Firefox, Konqueror, Opera, Safari, dan disupport juga oleh berbagai *platform*, seperti Microsoft Windows, UNIX/Linux, dan Mac OS X.

Karena tidak dapat diketahui *user* menggunakan *browser* apa, versi berapa, dan sistem operasi apa, maka kode yang ditulis harus dapat menyesuaikan dengan browser user sehingga *XMLHTTPRequest* dapat berjalan dengan sukses. Untuk sebagian besar *browser* ( Mozilla Firefox, Konqueror, Opera dan Safari ), dapat membuat *instant object* secara langsung, berikut adalah contoh pembuatan *variable* yang menyimpan hasil dari *XMLHTTPRequestobject.*

```
var request = new XMLHTTPRequest();
```

khusus untuk Microsoft Internet Explorer, harus dibuat *ActiveX object* terlebih dahulu, contohnya seperti di bawah ini.

```
var request = new ActiveXObject("Microsoft.XMLHTTP");
```

untuk mengatasi perbedaan versi MSXML Internet Explorer, perlu juga memberikan perintah seperti dibawah ini

var request = new ActiveXObject("Msxml2.XMLHTTP"); Solusi yang baik untuk masalah ini adalah dengan mencoba satu persatu *method* untuk membuat *instant object* sampai salah satu *method* berhasil.

Berikut adalah contoh perintah untuk mengatasi perbedaan *browser* dan versi dari Internet Explorer [Sams,2006].

**Tabel 2.2 XMLHttpRequest Object Properties**

| Property | Kegunaan | Read/write |
|---|---|---|
| **Onreadystatechange** | Menentukan event handler mana yang akan dipanggil ketika nilai object property ready state Berubah | Read/write |
| **readyState** | Laporan status permintaan dalam bentuk integer :<br><br>0 = uninitialized<br>1 = loading<br>2 = loaded<br>3 =interactive<br>4 =completed | Read-only |

| | | |
|---|---|---|
| responseText | Data yang dikirim oleh server dalam bentuk string | Read-only |
| **responseXML** | Data yang dikirim pleh server dalam bentuk XML | Read-only |
| **Status** | Status kode HTTP yang dikirim oleh server | Read-only |
| **statusText** | HTTP response status text yang dikirim oleh server | Read-only |

**Tabel 2.3 XMLHttpRequest Object Method**

| Method | Means |
|---|---|
| **abort()** | Menghentikan request yang sedang diproses |
| **getAllResponseHeaders()** | Pengembalian semua header dalam bentuk String |
| **getresponseHeader(x)** | Pengembalian nilai header x dalam bentuk string |

| | |
|---|---|
| **open('method','URL','a')** | Spesifikasi dari HTTP method (contohnya, GET atau POST), URL tujuan, dan ketika request ditangani secara asynchronously (JIka ya, a='TRue'the default; jika tidak, a='false'.) |
| **send(content)** | Mengirimkan request, dan dapat dilakukan secara bebas sesuai dengan *open method* yang digunakan |
| **setRequestHeader('x','y')** | Menentukan parameter dan nilai pasangan |

**Tabel 2.4 *Status Server* properties**

| **Property** | **Means** |
|---|---|
| **200** | Request telah sukses |
| **204** | Dokumen kosong |
| **401** | Request tidak otentik |
| **403** | Server menolak untuk memproses request |
| **404** | Request tidak ditemukan |
| **405** | Method tidak diijinkan |
| **408** | Client gagal mengirinkan request dalam selang waktu yang ditentukan oleh server |
| **500** | Server Error |
| **501** | Tidak diterapkan |

| | |
|---|---|
| **503** | Service tidak ditemukan |
| **505** | Versi HTTP tidak support |