Implementasi Jquery AJAX Untuk Fitur Pendataan Petani pada Website Dutatani

Wilfridus Bau Mau[1], Argo Wibowo[2], Rosa Delima[3]
Fakultas Teknologi Informasi, Universitas Kristen Duta Wacana
Jl. Dr. Wahidin Sudirohusodo no 5-25 Kotabaru, Yogyakarta
[1]wilfridus.mau@students.ukdw.ac.id,
[2]argo@staff.ukdw.ac.id,
[3]rosadelima@staff.ukdw.ac.id

***Abstract*— Dutatani is an integrated information system in the agricultural sector developed by a Duta Wacana Christian University team. Dutatani has several subsystems, including the farmer data collection system. The team developed a web-based farmer data collection system. This system was developed using web version 1.0 when requesting or receiving a response from the server. The system needs to carry out loading and reloading the page. The condition results in additional time for processing and loading the whole page. AJAX technology is part of the web generation 2.0. A website can make page changes without reloading the page through this technology. This feature can speed up the reload time of a web page. In this research, AJAX technology has been applied to improve the performance of the farmer data collection system. The system development stage includes literature study, design, development, and evaluation of the system prototype. Evaluation is carried out by comparing system performance before and after implementing AJAX technology. The research results show that AJAX technology can improve the performance of server requests by 5.2%. A significant difference in performance occurs in the amount of data transfer, where AJAX can increase the performance of the data transfer bit size by 86%.**

***Keywords*— Farmer Data Collection System, Dutatani Website, AJAX, Jquery, Efficiency reload page.**

Abstrak— Dutatani merupakan sebuah sistem informasi terintegrasi di bidang pertanian yang dikembangkan oleh tim dari Universitas Kristen Duta Wacana. Dutatani memiliki beberapa subsistem, salah satu diantaranya adalah sistem pendataan petani. Sistem pendataan petani dikembangkan berbasis web. Sistem ini dikembangkan menggunakan versi web 1.0, dimana ketika melakukan *request* atau menerima *response* dari *server*. Untuk memberikan respon, sistem melakukan proses *loading* dan *reload page*. Hal ini berdampak pada tambahan waktu untuk memproses dan memuat halaman secara penuh. Teknologi *AJAX* merupakan bagian dari web generasi 2.0. Melalui teknologi ini sebuah website dapat melakukan perubahan halaman tanpa harus melakukan *reload page*. Fitur ini dapat mempercepat waktu *reload* sebuah halaman web. Untuk meningkatkan *performance* pada sistem pendataan petani dilakukan pengembangan web dengan penerapan AJAX. Tahap pengembangan sistem meliputi studi pustaka, perancangan, pengembangan, dan evaluasi dari prototipe sistem. Evaluasi dilakukan dengan membandingkan performa sistem sebelum dan setelah menerapkan teknologi AJAX. Hasil penelitian menunjukan bahwa teknologi AJAX dapat meningkatkan performa waktu *request* ke *server* sebesar 5,2%. Perbedaan performa yang signifikan terjadi pada besaran *transfer* data dimana AJAX dapat meningkatkan performa besaran bit transfer data sebesar 86%.

***Kata Kunci*— Sistem Pendataan Petani, Website Dutatani, AJAX, Jquery, Efisiensi reload halaman.**

I. PENDAHULUAN

Dutatani merupakan sebuah sistem informasi terintegrasi di bidang pertanian. Sistem ini telah dikembangkan oleh Tim Fakultas Teknologi Informasi Universitas Kristen Duta Wacana (FTI UKDW) sejak tahun 2016 [1]. Salah satu sistem yang dikembangkan oleh Tim Dutatani adalah sistem pendataan petani. Sistem pendataan petani dikembangkan berbasis web dan menyatu dengan Website DutaTani [2][3][4]. Website dapat diakses pada alamat https://dutatani.id/.

Web yang digunakan untuk pengembangan sistem pendataan petani adalah web 1.0, dimana responsibilitas aplikasi berbasis web bergantung pada proses *loading* dan *reloading page*. Web 1.0 merupakan *website* yang digunakan untuk pertama kalinya, dimana teknologi ini memiliki halaman web statis yang perubahannya dapat terlihat ketika dilakukan *reload* halaman [5]. Misalkan pengguna ingin melakukan *login* pada sebuah halaman, jika terjadi kesalahan *username* atau *password* yang dimasukan, maka *server* harus *reloading* satu halaman penuh yang tujuannya hanya untuk menampilkan pesan kesalahan mengenai *username* dan *password* yang telah dimasukan, hal ini membuat *user* menunggu cukup lama untuk melihat pesan kesalahan. Pada aplikasi yang berbasis desktop, *user* dapat melihat langsung pesan jika terjadi kesalahan tanpa harus menunggu program menampilkan kembali seluruh halaman baru. Akan tetapi, interaksi seperti itu tidak dapat dilakukan pada aplikasi berbasis *web*. Pada aplikasi berbasis *web*, ketika *user* sudah memasukan data pada *form* yang telah disedikan, dengan menekan sebuah tombol *submit*, lalu *user* menunggu *browser* memuat sebuah halaman baru, dan sebagian besar dari halaman baru tersebut sama dengan halaman sebelumnya, seperti *header*, *sidebar*, *footer*, *background*, dan logo.

Pada penelitian ini, penulis akan mengimplementasikan AJAX pada *Website* Dutatani untuk pendataan petani. Dimana dalam *website* pendataan petani membutuhkan tampilan yang memungkinkan pengguna dapat melihat langsung perubahan data tanpa harus melakukan *reload*/ *refresh* halaman.

Artikel ini ditulis dalam empat bagian yaitu bagian pendahuluan, metode penelitian, hasil dan pembahasan, dan kesimpulan.

II. LANDASAN TEORI

Menurut penelitian [6] menyebutkan sejak dimulainya pada tahun 2004, era teknologi *website* modern mulai mengalami perubahan yang cukup signifikan, hal ini ditandainya dengan adanya kemunculan era versi *web* yang mengusung teknologi *web* 2.0 berbasiskan Rich Internet Application (RIA). Web 2.0 merupakan *website* generasi kedua, dimana memiliki halaman *web* yang perubahan datanya dapat terlihat tanpa harus dilakukan *reload* ulang halaman [7]. Dengan adanya teknologi *web* 2.0 aplikasi berbasis *web* sudah mencapai responsifitas yang lebih tinggi dibandingkan dengan web 1.0. *Web* 2.0 memilki beberapa kelebihan dibandingkan dengan *web* 1.0, misalnya untuk melakukan perubahan nama pada suatu data yang ada pada halaman *website*, pengguna tidak perlu melakukan *reload* seluruh halaman. *Browser* hanya melakukan perubahan pada data yang ingin diubah dan langsung menambahkan pada halaman yang sudah ada. Begitu juga saat *user* melakukan login, jika terjadi kesalahan *username* dan *password*, server dapat mengirimkan pesan kesalahan kepada *user* kedalaman halaman yang sedang terbuka tanpa melakukan *reload*/ *refresh page*.

AJAX merupakan sebuah teknologi yang dapat digunakan untuk membuat aplikasi-aplikasi yang dapat diterapkan pada web 2.0. Teknologi AJAX mampu berkomunikasi dengan *server* terus menerus untuk *request* informasi terbaru dan menampilkan di *browser client* tanpa harus melakukan proses *reload/refresh*. AJAX memiliki kemampuan dalam melakukan *load* sebuah halaman dan menampilkan data pada *browser* lebih cepat, dibandingkan pada *website* yang tidak menggunakan AJAX [8][9]. Hal ini dikarenakan AJAX menggunakan objek dari JavaScript yang ada pada *browser* yaitu XMLHttpRequest yang bisa berhubungan secara *asynchronous* yang artinya AJAX dapat melakukan *request* maupun menerima *response* dari *server* secara langsung tanpa adanya *reload* pada sebuah halaman *browser* [10].

AJAX menggunakan sebuah *interface* yang ada pada *browser* yang biasa disebut dengan istilah DOM (Document Object Model). DOM mampu menyeleksi elemen HTML dan CSS [11]. Elemen yang diseleksi dapat meliputi mengubah atribut dari sebuah elemen seperti atribut *src* pada *tag img* atau atribut *href* pada *tag a*, menambah dan menghapus elemen, dan melakukan aksi terhadap elemen melalui sebuah tombol atau *button trigger* [12]. AJAX dapat melakukan perubahan atau penambahan element DOM pada sebuah halaman *web* secara spesifik tanpa harus *reload* semua elemen yang ada [13]

Terdapat dua format pertukaran data yang dapat digunakan yaitu XML dan JSON. XML merupakan sebuah format data seperti *tag* HTML yang digunakan untuk mendiskripsikan data yang didefinisikan pada *tag* [14]. Sedangkan JSON merupakan sebuah format data yang berbasis *object* yang memiliki *value* saling berpasangan seperti *array* [15]. Menurut penelitian [16] mengatakan bahwa JSON dapat digunakan untuk pertukaran data secara ringan dan lebih cepat dibandingkan format XML.

III. METODE PENELITIAN

Penelitian ini berjalan dalam lima tahapan yaitu analisis Website Dutatani yang telah dikembangkan, perancangan penambahan Jquery AJAX pada *website*, implementasi, uji coba sistem, dan pengujian efektifitas dan efisien penerapan teknologi AJAX pada *website*. Tahapan penelitian dapat dilihat pada Gambar 1.

Gambar 1 Tahapan penelitian

A. Analisis Sistem

Pada tahap analisis sistem digunakan metode observasi untuk mengumpulkan data (fitur) pada *website* pendataan petani di Dutatani. Analisis sistem dilakukan dengan mengamati bagian/fitur dari sistem pendataan petani yang paling banyak melakukan *reload/refresh* halaman *web*. Tujuan untuk analisis ini adalah mengetahui dengan tepat dimana sebaiknya dilakukan penerapan teknologi AJAX kecepatan akses *website* dapat dioptimalkan.

Analisis sistem menghasilkan daftar fitur yang akan ditambahkan teknologi AJAX. Terdapat dua halaman utama pada *website* yang akan diterapkan teknologi AJAX yaitu halaman admin untuk pendataan petani dan halaman *user* petani. Pada halaman admin, teknologi AJAX akan diterapkan pada fitur tambah, ubah, hapus, dan pencarian data petani. Sementara pada halaman untuk *user* petani, teknologi AJAX diterapkan pada fitur ubah data, ganti *password*, validasi *password*, dan validasi *email*.

Setelah didapatkan halaman dan proses yang akan disematkan teknologi AJAX langkah selanjutnya adalah analisa elemen DOM yang akan digunakan. Penyeleksian elemen-elemen DOM berdasarkan selektor CSS yang memanfaatkan nama-nama dan atribut elemen, misalnya *id* dan *class*, sebagai kriteria seleksi elemen DOM.

B. Perancangan

Setelah dilakukan analisis dan ditentukan halaman web serta elemen DOM yang akan diterapkan Jquery AJAX, selanjutnya dilakukan perancangan perubahan sistem. Rancangan meliputi fungsi dan *method* yang akan dilakukan penerapan Jquery AJAX. Hasil dari proses perancangan dapat dilihat pada Tabel 1 dan Tabel 2. Tabel 1 merupakan rancangan penerapan Jquery AJAX pada Halaman Admin sementara Tabel 2 merupakan rancangan untuk Halaman Petani. Berdasarkan kedua tabel, terdapat 35 proses sistem yang akan disematkan teknologi Jquery AJAX. Perancangan membutuhkan *library* AJAX sehingga dapat memanggil fitur-fitur dan fungsi pemanggilan oleh Jquery AJAX. *Library* yang digunakan menggunakan AJAX terbaru sehingga dapat memanfaatkan semua fitur terbaru. AJAX akan mengakses URL HTTP pada fungsi-fungsi yang ada pada sistem sehingga perlu mengetahui juga semua HttpRequest yang ada pada sistem. Hal ini dapat diketahui dari URL *Route* sistem yang dapat dengan mudah diakses pada *class route*.

JUTEI Edisi Volume. 7 No.2 Oktober 2023
ISSN 2579-3675, e-ISSN 2579-5538
DOI 10.21460/jutei.72.270

TABEL 1
RANCANGAN PENERAPAN JQUERY AJAX UNTUK HALAMAN ADMIN

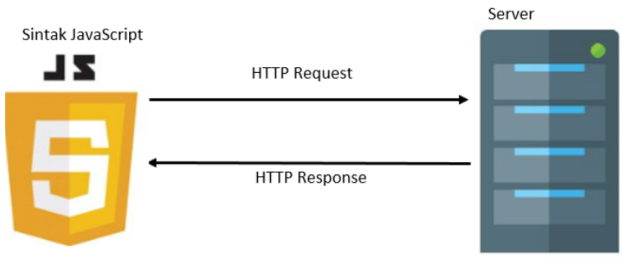
Fitur	**Rancangan fungsi dan method penerapan Jquery AJAX.**
Proses tambah data petani	Menambahkan *id* pada *tag form* dengan nama form_regis_oleh_admin, yang berfungsi agar tag form dapat dipanggil dan diambil datanya oleh *function submit*.
	Membuat *routes* baru dengan *url* register/petani/oleh/admin, yang berfungsi untuk menghubungkan *client-side* dengan sebuah *method* yang dituju yang ada pada *controller*.
	Membuat fungsi AJAX untuk mengirimkan atau menerima data dari *server*.
	Membuat *method* dengan nama register_petani_oleh_admin yang berfungsi untuk menerima *request* dari *client-side* kemudian memberi respon juga ke *client-side*.
Proses ubah data petani	Menambahkan fungsi *onclick* dengan nama ubah_petani yang memiliki 1 parameter yaitu *id* petani pada *button* ubah petani
	Membuat *routes* baru dengan url admin/ubah/petaniajax, yang berfungsi untuk menghubungkan *client-side* dengan sebuah *method* yang dituju yang ada pada *controller*
	Membuat *function javascript* dengan nama ubah_petani yang memiliki 1 parameter yang berfungsi menyimpan fungsi *AJAX*.
	Membuat fungsi *AJAX* untuk mengirimkan atau menerima data dari *server*.
	Membuat method dengan nama *fetch*_ubah_petani, yang berfungsi untuk menerima request dari *client-side* kemudian memberi respon juga ke *client-side*.
Proses hapus data petani	Menambahkan fungsi *onclick* dengan nama hapus_petani yang memiliki 1 parameter yaitu *id* petani pada *button* hapus petani
	Membuat routes baru dengan url admin/hapus/petani/ajax, yang berfungsi untuk menghubungkan client-side dengan sebuah *method* yang dituju yang ada pada *controller*
	Membuat function *javascript* dengan nama hapus_petani yang memiliki 1 parameter yang berfungsi menyimpan fungsi *AJAX*.
	Membuat fungsi *AJAX* untuk mengirimkan atau menerima data dari server.
	Membuat method dengan nama hapus_petani_ajax, yang berfungsi untuk menerima *request* dari *client-side* kemudian memberi respon juga ke *client-side*.
Proses pencarian data petani	Menambahkan fungsi *onkeyup* pada masukan *search*, yang berfungsi untuk menangkap langsung kata kunci pencarian
	Membuat *function* dengan nama *fetch_data* memiliki empat parameter, yang berfungsi untuk mengirimkan *request* kata kunci dan mendapatkan respon secara langsung dari *server*
	Membuat *routes* baru dengan *url /fetch_list_petani*, yang berfungsi untuk menghubungkan *client-side* dengan sebuah method yang dituju yang ada pada *controller*
	Membuat *method* dengan nama *fetch_list_petani*, yang berfungsi untuk menerima *request* dari *client-side* kemudian memberi respon juga ke *client-side*.

TABEL 2
RANCANGAN PENERAPAN JQUERY AJAX UNTUK HALAMAN PETANI

Fitur	**Rancangan fungsi dan method penerapan *Jquery AJAX*.**
Proses ubah data diri petani	Menambahkan *id* pada *tag form* dengan nama upload_form_ubah_petani, yang berfungsi agar tag form dapat dipanggil dan diambil datanya oleh *function submit*.
	Membuat *routes* baru dengan url /update/profile/petani/aksi/ajax, yang berfungsi untuk menghubungkan *client-side* dengan sebuah *method* yang dituju yang ada pada *controller*.
	Membuat fungsi *AJAX* untuk mengirimkan atau menerima data dari *server*.
	Membuat *method* dengan nama update_profile_petani_ajax, yang berfungsi untuk menerima *request* dari *client-side* kemudian memberi respon juga ke *client-side*.
Proses ganti password petani	Menambahkan fungsi *onclick* dengan nama *update_password* pada *button update password*
	Membuat *routes* baru dengan url /update/passwordajax, yang berfungsi untuk menghubungkan client-side dengan sebuah *method* yang dituju yang ada pada *controller*.
	Membuat *function javascript* dengan nama *update_password* yang memiliki 1 parameter yang berfungsi menyimpan fungsi *AJAX*.
	Membuat fungsi *AJAX* untuk mengirimkan atau menerima data dari *server*.
	Membuat *method* dengan nama *update_password_ajax*, yang berfungsi untuk menerima *request* dari *client-side* kemudian memberi respon juga ke *client-side*.
Proses validasi ganti password petani	Menambahkan fungsi *keyup* pada masukan konfirmasi *password*, yang berfungsi agar dapat membaca secara langsung masukan untuk divalidasi *server*.
	Membuat *routes* baru dengan *url/cek/password*, yang berfungsi untuk menghubungkan *client-side* dengan sebuah method yang dituju yang ada pada *controller*.
	Membuat fungsi *AJAX* untuk mengirimkan atau menerima data dari *server*.
	Membuat *method* dengan nama *cek_password*, yang berfungsi untuk menerima *request* dari *client-side* kemudian memberi respon juga ke *client-side*.
Proses validasi email pada halaman ubah data diri petani	Menambahkan fungsi *onkeyup* pada masukan *email*, yang berfungsi untuk rekam masukan dari pengguna
	Membuat *routes* baru dengan *url/cek/email*, yang berfungsi untuk menghubungkan *client-side* dengan sebuah *method* yang dituju yang ada pada *controller*.
	Membuat fungsi *AJAX* untuk mengirimkan atau menerima data dari *server*.
	Membuat *method* dengan nama *cek_email*, yang berfungsi untuk menerima *request* dari *client-side* kemudian memberi respon juga ke *client-side*.

C. Penerapan AJAX pada fitur pendataan petani

Pada tahap ini hal yang dilakukan adalah melakukan implementasi fungsi AJAX yang telah dirancang untuk setiap fitur. Penerapan Jquery AJAX diawali dengan pembuatan *request* pada *Client Side* seperti yang tampak pada gambar 2.

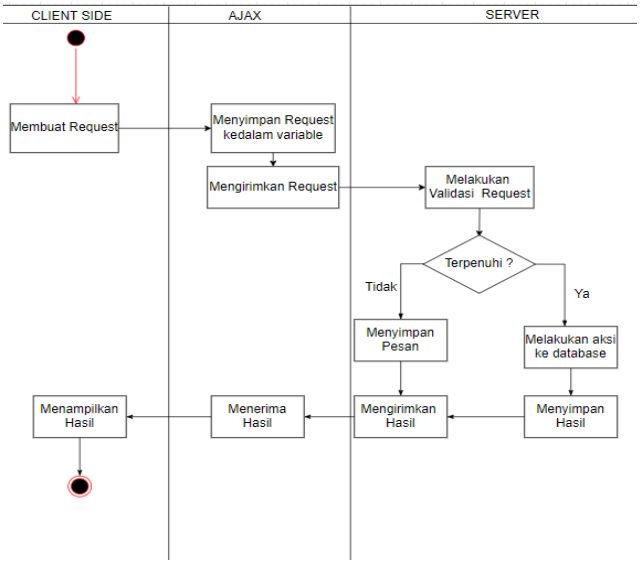


Gambar 2 Ilustrasi Teori AJAX

Setelah pembuatan *request* (Gambar 2), selanjutnya *AJAX* akan menyimpan dan mengirim *request*. Pada bagian *server* akan dilakukan validasi terhadap *request*. Jika *request*

dapat dijalankan maka akan dilakukan pemrosesan pada basis data, dan menyimpan hasil yang selanjutnya dikirim ke AJAX dan ditampilkan pada *Client Side*. Diagram proses penerapan AJAX dapat dilihat pada Gambar 3. Untuk pengkodean program, penerapan *Jquery AJAX* dilakukan dengan penambahan baris kode dari pada aplikasi yang telah dikembangkan. Sebagai contoh dapat dilihat pada Gambar 4, untuk baris program pada fitur tambah data petani.

Pada Gambar 4 dapat dilihat bahwa elemen DOM dibutuhkan untuk dapat mendeteksi objek yang ada pada sistem. Setelah terdeteksi maka dapat dibuat fungsi yang memanggil *HttpRequest* pada sistem. Fungsi dibuat pada kondisi *submit* sehingga hanya dapat dijalankan ketika pengguna melakukan klik *submit* pada elemen tertentu yang telah ditentukan. Tipe metode yang digunakan juga disesuaikan dengan metode yang ada sebelumnya pada sistem. Tipe data yang digunakan ada JSON yaitu tipe data yang saat ini banyak digunakan sebagai jembatan antar sistem. Karena tipe data yang dikirim JSON maka perlu adanya proses pembentukan data dalam format JSON sebelum dikirimkan melalui AJAX. Pada sistem sebelumnya format JSON sudah digunakan untuk mengirimkan data dari antarmuka kepada *controller* sehingga pengembangan sistem menggunakan AJAX tidak sulit untuk dilakukan.



Gambar 3 Diagram penerapan AJAX

```
$(document).ready(function(){
  $('#form_regis_oleh_admin').on('submit', function(e){
      e.preventDefault();
          $.ajax({
              url : '/register/petani/oleh/admin',
              type : 'post',
              data : new FormData(this),
              dataType : 'json',
              contentType: false,
              cache: false,
              processData: false,
              success : function(data){
              }
          });
      });
  });
```

Gambar 4 Contoh kode program untuk penerapan *Jquery AJAX* pada fitur penambahan data petani.

D. Pengujian

Pada tahap ini akan dilakukan pengujian menggunakan properti yang ada pada *browser* yaitu *Performance Navigation Type* dan *Performance Navigation Type Reload*. Kedua properti nantinya akan disisipkan pada baris program atau fungsi dari fitur yang diimplementasikan Jquery AJAX. Gambar 5 merupakan baris kode untuk pengecekan halaman *reload*, sementara Gambar 6 merupakan fungsi *log* untuk *testing*. Fungsi *log testing* digunakan untuk hasil uji coba dari penerapan *Jquery* AJAX.

Pengecekan halaman *reload* dilakukan secara berkala, terlebih apabila sistem memiliki percabangan dalam pemeriksaan variabel. Jika percabangan terlalu banyak atau lebih dari 1 maka digunakan *case* untuk menyederhanakan pemeriksaan percabangan. Di setiap percabangan telah disiapkan variabel yang menampung pesan *log* sesuai dengan tugas dari percabangan tersebut. Upaya ini untuk mendapatkan data yang akurat apakah *website* melakukan *reload* atau terjadi kesalahan pada saat pemeriksaan variabel pada percabangan tertentu. Pada penelitian ini menggunakan metode *GET* pada pemanggilan fungsi *website* karena tidak diperlukan pengiriman parameter khusus.

```
function check_reload(fitur){
    let info = "";
    switch (fitur) {
      case 'p_ubah_data':
        info = "Ubah data petani";
        break;
      case 'p_ganti_password':
        info = "Ganti password petani";
        break;
      case 'p_validasi_password':
        info = "Validasi password petani";
        break;
      case 'p_validasi_email':
        info = "Validasi email petani";
        break;
    }
    if(performance.navigation.type == performance.navigation.TYPE_RELOAD){
      info = "Halaman "+ info + " Melakukan Reload";
    }else{
      info = "Halaman " + info + " Tidak Melakukan Reload";
    }

    $.ajax({
      url : '/logTesting',
      type : 'get',
      data : {'log' : info},
      dataType : 'json',
      success : function(data){}
    });
}
```

Gambar. 5 Fungsi Cek *Reload*

```
public function log_testing(Request $request){
    $info = $request->get('log');
    Log::info('Testing '.$info);
}
```

Gambar 6 Fungsi *Log* Testing

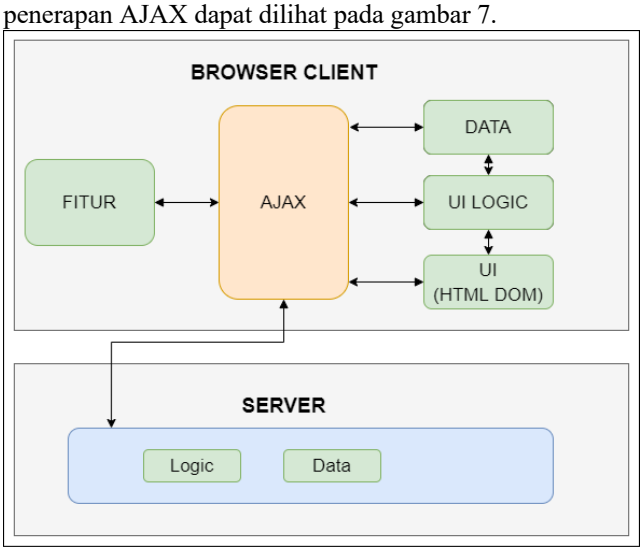
IV. HASIL DAN PEMBAHASAN

A. Arsitektur Sistem

Secara sederhana, arsitektur fitur AJAX terdiri dari dua bagian yaitu *browser client* dan *server*. Pada sisi *client* terdapat fitur sistem yang dikembangkan. *Client* akan mengirim *request* dan AJAX akan menangani *request* tersebut. Jika dalam proses *request* membutuhkan *logic* di sisi *client* maka *Jquery* langsung melakukan proses langsung pada sisi *client*, namun jika proses membutuhkan data atau *logic* dari sisi *server*, maka AJAX langsung meminta *request* ke *server* dan memprosesnya. Selanjutnya setelah *request* didapatkan maka *server* langsung mengirimkan *response* melalui ajax untuk ditampilkan disisi *client*. Arsitektur

JUTEI Edisi Volume. 7 No.2 Oktober 2023
ISSN 2579-3675, e-ISSN 2579-5538
DOI 10.21460/jutei.72.270

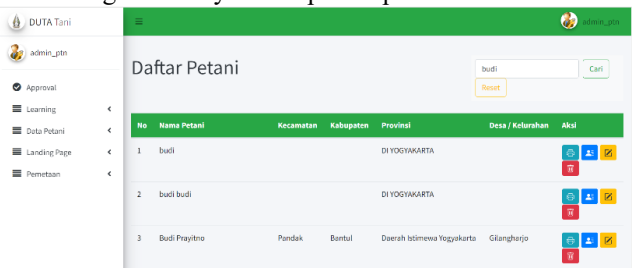
penerapan AJAX dapat dilihat pada gambar 7.



Gambar 7 Arsitektur penerapan AJAX

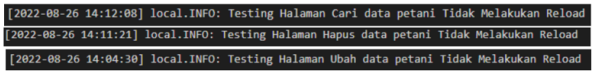
B. Hasil Uji Coba

Penelitian ini berfokus pada bagian pengembangan akses data pada fungsi cek *reload*. Oleh karena itu pengembangan besar pada antarmuka tidak dilakukan. Perubahan dilakukan pada level DOM sehingga tidak diperlukan pembaharuan antarmuka. Elemen DOM sudah didefinisikan sejak awal pengembangan sistem dengan tujuan penggunaan *javascript* dan persiapan pengembangan sistem jangka panjang. Elemen DOM yang sudah ada dimanfaatkan untuk menjaga konsistensi sistem yang sudah ada sebelumnya. Hal yang paling terlihat dari pemanfaatan AJAX dan DOM adalah halaman yang tidak mengalami *reload* seperti yang terlihat pada gambar 8. Ketika melakukan pencarian pada seorang pengguna pada kotak pencarian, maka secara *realtime* sistem dapat mengambil data dan menampilkannya tanpa melakukan proses *reload browser*. Teks pada kotak pencarian tetap ada setelah proses pencarian. Hal ini memudahkan pengguna ketika melakukan pencarian dan ingin mengingat apa yang sudah dicari pada proses sebelumnya. Pada sistem sebelumnya ketika melakukan proses pencarian maka isian kotak pencarian akan hilang dan menyulitkan proses pencarian.



Gambar 8 Perubahan Proses Data Antarmuka

Selanjutnya untuk mengetahui apakah sistem berjalan atau tidak maka diberi *log* pada sistem untuk memberi *trigger* apabila sistem sukses melakukan pemanggilan AJAX. Apabila sistem sukses melakukan pemanggilan AJAX maka sistem tidak perlu melakukan *reload* untuk memanggil data. Pada gambar 9 dapat dilihat *log* berhasil muncul ketika sebuah pemanggilan AJAX berhasil dilakukan pada beberapa proses sistem. Upaya ini dilakukan untuk mempermudah pembuktian upaya penerapan AJAX pada sistem, karena seperti yang tampak pada Gambar 8 perubahan pada antarmuka tidak terlalu terlihat.



Gambar 9 Bukti *Log* Sistem

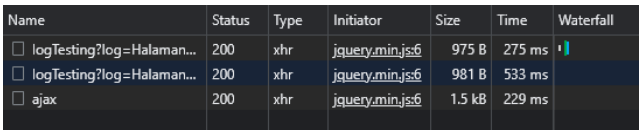
Pengujian dilakukan untuk setiap fitur yang didefinisikan pada Tabel 1 dan Tabel 2. Uji coba dilakukan dengan menguji fitur yang dikembangkan dan menambahkan fungsi cek *reload*. Fungsi ini ditujukan untuk mengecek apakah terjadi *reload* halaman *web* atau tidak pada saat diaktifkan fitur tertentu. Setelah mendapat hasil dari fungsi cek *reload* maka hasil akan disimpan di *log testing* pada Laravel. Rangkuman hasil uji coba sistem dapat dilihat pada Tabel 3.

TABEL 3
HASIL TESTING RELOAD

No	Fitur	Hasil yang diharapkan	Hasil pengujian
1	Proses tambah data petani	Tidak melakukan *reload* halaman	**Berhasil**
2	Proses ubah data petani	Tidak melakukan *reload* halaman	**Berhasil**
3	Proses hapus data petani	Tidak melakukan reload halaman	**Berhasil**
4	Proses pencarian data petani	Tidak melakukan *reload* halaman	**Berhasil**
5	Proses ubah data diri petani	Tidak melakukan *reload* halaman	**Berhasil**
6	Proses ganti password petani	Tidak melakukan *reload* halaman	**Berhasil**
7	Proses validasi ganti password petani	Tidak melakukan *reload* halaman	**Berhasil**
8	Proses validasi email	Tidak melakukan *reload* halaman	**Berhasil**

C. Analisis Efisiensi

Analisis efisiensi penerapan Jquery AJAX dilakukan melalui analisis waktu dan besaran transfer data. Pengujian dilakukan dengan membandingkan antara waktu dan besaran transfer data sebelum dan setelah diterapkan *Jquery AJAX*. Untuk analisis waktu ini digunakan fungsi yang ada pada *browser* yaitu *inspect element* pada bagian *network* seperti pada Gambar 10.



Gambar 10 Waktu Response Server

Pada Gambar 10, dapat dilihat waktu proses *request* dan *response* ke *server* pada setiap fitur yang telah diimplementasikan Jquery AJAX. Sementara itu, hasil perhitungan waktu untuk setiap fitur sebelum dan setelah penerapan *Jquery AJAX* dapat dilihat pada Tabel 4.

Berdasarkan hasil pengujian pada Tabel IV, dapat dilihat bahwa rata-rata waktu reload halaman tanpa AJAX adalah sebesar 405ms, sementara waktu reload halaman setelah diterapkan Jquery AJAX sebesar 384ms. Untuk mengetahui apakah terjadi efisiensi waktu *reload* halaman, dilakukan analisis statistik uji beda. Hasil uji beda dengan signifikan level 5%, menunjukan bahwa rata-rata waktu setelah diimplementasikan AJAX berbeda dengan rata-rata waktu

sebelumnya. Hal ini berarti penerapan *Jquery AJAX* dapat meningkatan efisiensi waktu reload halaman web untuk pendataan petani sebesar 5,2%.

TABEL 4
ANALISIS WAKTU PENERAPAN AJAX

No	Fitur	Waktu Tanpa AJAX	Waktu Dengan AJAX
1	Proses tambah data petani	248ms	218ms
2	Proses ubah data petani	514ms	616ms
3	Proses hapus data petani	236ms	230ms
4	Proses pencarian data petani	678ms	475ms
5	Proses ubah data diri petani	122ms	218ms
6	Proses ganti password petani	415ms	403ms
7	Proses validasi ganti password petani	360ms	410ms
8	Proses validasi email	670ms	503ms
Rata-rata		**405ms**	**384ms**

Analisis efisiensi yang kedua dilakuan pada besaran transfer data untuk *reload* halaman. Sama seperti analisis waktu, analisis ini juga dilakukan dengan menggunakan dari fungsi yang ada pada *browser* yaitu *inspect element* pada bagian *network* pada Tabel 4. Analisis dilakukan dengan membandingkan rata-rata besaran transfer data sebelum dan setelah diterapan *Jquery AJAX*. Hasil analisis dapat dilihat pada Tabel 5.

TABEL 5
ANALISIS BESARAN TRANSFER DATA

No	Fitur	Besaran Transfer Data (kB)	
		Tanpa AJAX	Dengan AJAX
1	Proses tambah data petani	21.4	0.981
2	Proses ubah data petani	57.6	1.6
3	Proses hapus data petani	57.6	0.987
4	Proses pencarian data petani	39	29.5
5	Proses ubah data diri petani	23	1.5
6	Proses ganti password petani	16.4	1.0
7	Proses validasi ganti password petani	16.4	0.989
8	Proses validasi email	15.3	0.996
Rata-rata		**30**	**4.6**

Hasil evaluasi pada Tabel 5 menunjukan perbedaan yang sangat signifikan pada rata-rata besaran transfer data dalam kilobite (kB), dimana sebelum diterapan Jquery AJAX didapat rata-rata 30kB dan terjadi penurunan besaran tranfer data sebesar 4,6kB. Hal ini berarti penerapan Jquery AJAX mampu meningkatkan efisiensi besaran transfer data sekitar 86%.

V. KESIMPULAN

Kesimpulan yang dapat dirumuskan pada penelitian ini, sebagai berikut: (1) Penerapan Jquery AJAX pada sebuah suatu aplikasi menghasilkan proses perubahan data pada halaman *web* secara langsung dari *server* ke *browser client*, tanpa harus melakukan proses *reload/refresh*. Hal ini membawa perubahan perilaku sistem dan menjaga konsistensi serta navigasi pengguna dalam menggunakan sistem; (2) Penerapan *Jquery AJAX* pada *website* Dutatani untuk pendataan petani, berdampak pada peningkatan efisiensi waktu sebesar 5,2% dan penurunan besaran transfer data sekitar 86%; dan (3) Pemanggilan *HttpRequest* hanya menggunakan metode *GET* karena belum membutuhkan parameter khusus dalam pengujian. Perlu adanya perubahan menggunakan metode *POST* agar mendukung pengembangan sistem jangka panjang dan dapat mengirimkan variabel kepada fungsi sistem dengan lebih baik dan aman.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Fakultas Teknologi Informasi Universitas Kristen Duta Wacana yang telah memberi kesempatan bagi kami untuk melaksanakan penelitian dan penerbitan artikel ini.

REFERENSI

[1] A. R. Chrismanto, A. Wibowo, H. Budi, R. Delima, and J. Purwadi, "Menuju Pertanian Presisi dengan Sistem Pertanian Terintegrasi," in *Seminar Nasional "Peran Peneliti Perguruan Tinggi dalam Pembangunan Berkelanjutan"*, 2020, no. November, pp. 21–49.

[2] R. Delima, H. B. Santosa, and J. Purwadi, "Development of Dutatani Website Using Rapid Application Development," *IJITEE (International J. Inf. Technol. Electr. Eng.*, vol. 1, no. 2, pp. 36–44, 2017, doi: 10.22146/ijitee.28362.

[3] A. Wibowo, H. B. Santoso, R. Delima, and A. Rachmat, "Pengujian Usabilitas Portal Dutatani Menggunakan Metode Webqual Dan Importance Performance Analysis (IPA)," in *Seminar Nasional Sains Dan Teknologi 2019*, 2019, no. July, pp. 1–12.

[4] R. Delima, A. R. Chrismanto, A. Wibowo, and H. B. Santoso, "HILIRISASI SISTEM PEMETAAN LAHAN PERTANIAN TERINTEGRASI "DUTATANI" BAGI KELOMPOK TANI DI DESA GILANGHARJO BANTUL," in *Sendimas VI Tahun 2021*, 2021, pp. 16–25.

[5] R. Mubarok and N. N. Afifah S, "Artikel Sejarah Web Service," *www.researchgate.net*, 2020. https://www.researchgate.net/publication/339663250

[6] A. B. Warsito, A. Ananda, and D. Triyanjaya, "Penerapan Data JSON Untuk Mendukung Pengembangan Aplikasi Pada Perguruan Tinggi Dengan Teknik Restfull Dan Web Service," *Technomedia J.*, vol. 2, no. 1, pp. 26–36, 2017, doi: 10.33050/tmj.v2i1.313.

[7] Hendri, "Pemanfaatan Elearning Dengan Aplikasi Web 2.0 Sebagai Sarana Pembelajaran Pada Perguruan Tinggi Di Indonesia," *J. Ilm. Media Process.*, vol. 9, no. 2, pp. 150–156, 2014.

[8] M. A. Tahir, "Implementasi Ajax Pada Aplikasi Index Artikel Berbasis Web," *J. Ilm. Sist. Inf. dan Tek. Inform.*, vol. 1, no. 2, pp. 60–68, 2018.

[9] T. Wijaya, "Penerapan AJAX dalam Aplikasi Mobile Berbasis Web Untuk Meningkatkan Efisiensi Bandwidth," *Techno.Com*, vol. 17, no. 2, pp. 197–207, 2018, doi: 10.33633/tc.v17i2.1687.

[10] E. T. Uspandi and H. Witriyono, "Implementasi Proteksi Jquery Ajax Dengan Proteksi Sesion Pada Pengembangan Sistem Informasi Manajemen Kuliah Kerja Nyata di Universitas Muhammadiyah Bengkulu," *J. Media Infotama*, vol. 17, no. 2, pp. 45–52, 2021, [Online]. Available: https://jurnal.unived.ac.id/index.php/jmi/article/view/1645

[11] R. Trimanadi, F. Adikara, and H. S. Sastramihardja, "Usulan Pemanfaatan AJAX untuk Membangun Rich Internet Application pada Aplikasi Smartcity," in *Konferensi Nasional Sistem Informasi 2018*, 2018, pp. 8–9.

[12] E. P. Wijaya, "Berkenalan Dengan DOM (Document Object Model)," *Www.Gamelab.Id*, 2021. https://www.gamelab.id/news/857-berkenalan-dengan-dom-document-object-model (accessed Jun. 16, 2022).

[13] A. Nugroho, "Mengenal JQuery: Pengertian, Fungsi Dan Fiturnya," *Qwords*, 2020. https://qwords.com/blog/jquery-adalah/ (accessed Jun. 16, 2022).

[14] R. Risyan, "Apa Perbedaan JSON Dan XML? - Monitor Teknologi," *Monitor Teknologi*, 2020. https://www.monitorteknologi.com/perbedaan-json-dan-xml/ (accessed Jun. 16, 2022).

JUTEI Edisi Volume. 7 No.2 Oktober 2023
ISSN 2579-3675, e-ISSN 2579-5538
DOI 10.21460/jutei.72.270

[15] Nauval, "JavaScript: 10 Menit Dengan XMLHttpRequest & AJAX | by Nauval | Medium," *Medium.com*, 2019. https://medium.com/@mhdnauvalazhar/javascript-10-menit-dengan-xmlhttprequest-ajax-616128b9226 (accessed Jun. 16, 2022).

[16] M. A. Rosid, "Implementasi JSON untuk Minimasi Penggunaan Jumlah Kolom Suatu Tabel Pada Database PostgreSQL," *JOINCS (Journal Informatics, Network, Comput. Sci.*, vol. 1, no. 1, p. 33, 2017, doi: 10.21070/joincs.v1i1.802.