Rancang Bangun Aplikasi Pembuatan Presentasi Berbasis Web Menggunakan HTML5

Prima Astiadi[1], Oesman Hendra Kelana[2]
[1,2]Program Studi Teknik Informatika, Universitas Ma Chung
Jl. Villa Puncak Tidar N-01, Malang 65151
Email: [1]primaastiadi@gmail.com, [2]oesman.hendra@machung.ac.id

Abstract. *The goal of this thesis is to simplify the process of creating a web-based presentation and to provide solutions for compatibility problems in a standalone application. HTML5 is the latest web technology which is constantly developed and will be the successor of the previous HTML generation. This service for creating web-based presentation will make use of HTML5 features and JavaScript to deliver a user-friendly web-based application, with a concern in receiving text as its input and maintening it into a web-based presentation.This web-based presentation-making application, namely Slides5, provides an interface for creating a presentation with some features such as the addition and removal of slides, changing the look and layout of slides, as well as save and download the presentation file. Through several testings, Slides5 has succeeded in creating web-based presentation materials without the need of programming. In addition, Slides5 provides a feature to save the presentation as a HTML file so that the presentation can be simply concucted by using any web browser.*
Keywords: *web-based presentation, HTML5, Slides5*

Abstrak. *Tujuan dari penelitian ini adalah menyederhanakan proses pembuatan presentasi berbasis webdan memberikan solusi atas permasalahan kompatibilitas pada aplikasi mandiri.HTML5 merupakan teknologi web terbaru yang terus dikembangkan dan akan segera menjadi penerus dari generasi HTML sebelumnya. Layanan pembuatan presentasi berbasis web yang dimaksud akan memanfaatkan fitur HTML5 dan JavaScript untuk menghadirkan aplikasi berbasis web yang ramah pengguna, dengan menitikberatkan dalam menerima masukan berupa teks dan mengelolanya menjadi presentasi berbasis web.Aplikasi pembuatan presentasi berbasis web yang bernama Slides5 ini menyediakan antarmuka pembuatan presentasi dengan fitur-fitur seperti penambahan dan penghapusan slide, pengubahan tampilan dan tata letak slide, serta penyimpanan dan pengunduhan berkas presentasi. Melalui sejumlah uji coba, Slides5 telah berhasil dalam membuat materi presentasi berbasis web tanpa harus melakukan pemrograman. Selain itu, Slides5 menyediakan fitur penyimpanan presentasi ke dalam bentuk berkas HTML sehingga presentasi dapat dilakukan hanya dengan menggunakan peramban web saja.*
Kata Kunci: *presentasi berbasis web, HTML5, Slides5*

1. Pendahuluan

1.1. Latar Belakang

Dewasa ini, aktivitas presentasi menjadi sebuah kegiatan yang banyak dilakukan oleh berbagai kalangan dengan tujuan edukasi, persuasi maupun pemberian informasi. Materi presentasi pada umumnya dibuat dengan penyajian yang begitu menarik yang mencakup hal-hal yang akan dijelaskan oleh pemberi materi.

Kehadiran aplikasi *standalone* seperti Microsoft PowerPoint, Libre Office Impress, ataupun Key Note pada sistem operasi yang berbeda sangat membantu dalam menyajikan presentasi yang menarik serta interaktif. Namun pada kenyataannya, setiap aplikasi tersebut memiliki keunikan masing-masing, terutama jenis ekstensi berkas yang berbeda untuk tiap-tiap aplikasi. Hal ini menimbulkan masalah kompatibilitas yang menyebabkan berkas yang tercipta dari sebuah aplikasi pembuat presentasi hanya dapat dibuka dan ditampilkan dengan sempurna oleh aplikasi pembuat presentasi itu sendiri. Sebagai contoh, presentasi yang dibuat dengan menggunakan PowerPoint hanya dapat ditampilkan dengan sempurna apabila menggunakan

PowerPoint itu sendiri. Permasalahan tersebut juga berlaku pada presentasi yang dibuat dengan aplikasi berbeda. Beberapa aplikasi pembuat presentasi telah memberikan fitur untuk menyimpan berkas ke dalam format yang dapat dimengerti oleh aplikasi pembuat presentasi sejenis, namun hal tersebut masih belum dapat menghindarkan perubahan format yang terjadi saat berkas presentasi tersebut dibuka menggunakan aplikasi lain.

Kehadiran layanan berbasis *web* seperti Google Docs membawa alternatif solusi dengan kemampuannya untuk menyajikan konten presentasi secara *online*. Dengan terlebih dahulu terkoneksi dengan internet, seseorang dapat menampilkan berkas presentasi yang telah dibuat sebelumnya menggunakan Google Docs. Google Docs juga memberikan fitur untuk mengunduh berkas yang telah dibuat, namun fitur penyimpanan berkas masih terbatas pada ekstensi PDF, PPT, TXT dan format berupa gambar seperti JPEG, PNG, dan SVG. Tentu saja permasalahan kompatibilitas masih belum dapat dihilangkan sepenuhnya.

Berdasarkan permasalahan di atas, tercipta ide untuk menyediakan sebuah layanan pembuatan presentasi berbasis *web* yang dapat dengan mudah digunakan oleh semua kalangan melalui peramban *web* yang dapat ditemui di berbagai jenis sistem operasi. Nantinya, peramban *web* digunakan sebagai media untuk membuat dan menayangkan presentasi yang telah dibuat. Selain itu, layanan ini akan menciptakan berkas presentasi dalam format berkas HTML (*HyperText Markup Language*) yang dapat diterjemahkan oleh mayoritas peramban *web* yang ada. Kehadiran layanan pembuatan presentasi berbasis *web* ini diharapkan dapat menjadi sebuah alternatif untuk dapat mengatasi masalah kompatibilitas dan penyajian presentasi yang pada umumnya ditemui di aplikasi-aplikasi *standalone* untuk membuat presentasi yang banyak beredar.

1.2. Perumusan Masalah

Berikut adalah beberapa pokok permasalahan yang ada: (1) Bagaimana membuat layanan presentasi melalui *web* yang dapat membantu pengguna untuk membuat presentasi tanpa harus berurusan dengan kode-kode pemrograman terkait. (2) Bagaimana memberikan hasil layanan yang dapat mengatasi permasalahan kompatibilitas ekstensi berkas presentasi yang pada umumnya ditemukan pada aplikasi *standalone* untuk membuat presentasi.

1.3. Tujuan Penelitian

Tujuan umum dari pembuatan layanan ini adalah untuk menyediakan aplikasi berbasis *web* yang dapat berfungsi sebagai alat bantu untuk membuat presentasi. Sedangkan, tujuan khusus dari layanan yang ingin dikembangkan adalah sebagai berikut: (1) Menyediakan layanan pembuatan presentasi berbasis *web* yang dapat diterima oleh semua kalangan pengguna internet, baik pengguna dengan latar belakang IT maupun bukan. (2) Memberikan alternatif solusi presentasi melalui *web* untuk mengatasi permasalahan kompatibilitas yang ditemui pada aplikasi *standalone*.

2. Tinjauan Pustaka

2.1. HTML5

HTML adalah sebuah bahasa *markup* yang menjadi pusat dari struktur desain sebuah *website* (Powell, 2010). Kode yang berada di belakang setiap tampilan halaman *website* ini memungkinkan pengembang *website* untuk menciptakan tampilan dari sebuah halaman *website* sesuai dengan yang diinginkannya. Dikembangkan pertama kali oleh Tim Berners-Lee pada tahun 1989, HTML menjadi bahasa *markup* yang terus berkembang hingga saat ini (David, 2010). Salah satu hal yang menyebabkan HTML menjadi bahasa yang sukses dalam perkembangannya adalah konsep sederhana akan penggunaan *tag* yang menandai awal maupun akhir dari sebuah bagian. Hingga saat ini, HTML4 menjadi bahasa *markup* standar yang digunakan untuk mengembangkan *website*.

Pada tahun 2007, sebuah tim yang bernama Web Standards Project memulai untuk membuat generasi penerus dari HTML, yaitu HTML generasi ke 5 (David, 2010). HTML5 menyertakan pendefinisian ulang dari element *markup* yang ada dan pendefinisian elemen-

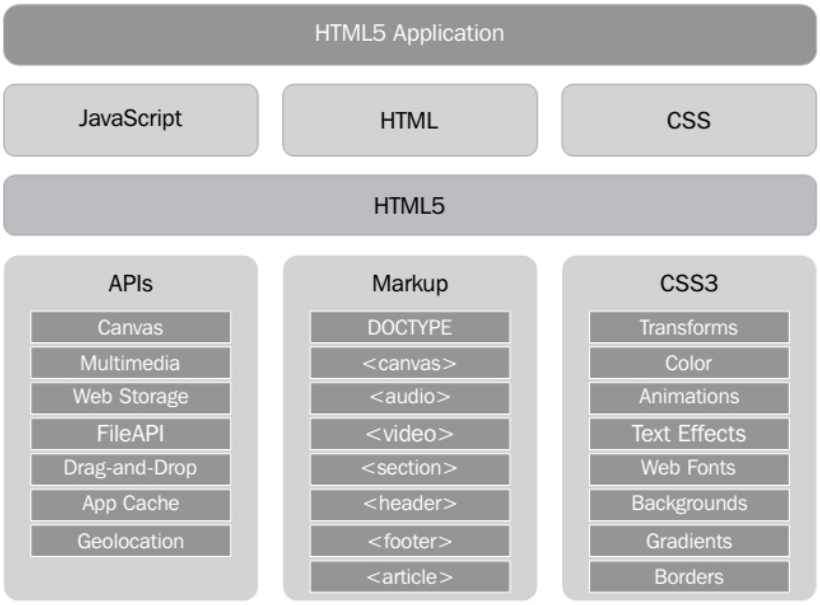
elemen baru yang memungkinkan para pendesain *web* untuk lebih ekspresif dalam semantik dari *markup*-nya (Glodstein, 2011). Dengan tujuan memberikan standar yang memungkinkan untuk menjalankan aplikasi dalam sebuah peramban *web*, HTML5 memperkenalkan beberapa fitur-fitur baru, beberapa diantaranya adalah: Elemen HTML baru, *Geolocation* APIs (*Application Programming Interface*), *Localdata* APIs, *Forms* 2.0., dukungan untuk video dan audio, CSS3, pencitraan gambar 2 dimensi dan 3 dimensi, JavaScript 2.0.

HTML5 kompatibel dengan versi-versi sebelumnya. Semua spesifikasi fitur HTML4 ada dalam HTML5, bahkan mengalami perubahan dan pengembangan (Clark, 2012). Perbedaan utama HTML5 dan HTML 4 adalah adanya tambahan API (*Application Programming Interfaces*) Javascript dalam spesifikasinya. Tambahan API ini memampukan adanya interaksi dengan elemen multimedia di dalam peramban (Devlin, 2012). HTML5 juga mengembangkan antarmuka pengguna (*user interface*), kemampuan akses (*accessability*), efek visual (*visual effect*), multimedia yang semakin tidak bergantung pada *plug-ins*, dan *web sockets* yang menjadikan koneksi dengan *web server* menjadi lebih persisten (Hogan, 2013).

Seperti pendahulunya, HTML5 dirancang sebagai bahasa yang *cross-platform*, penggunaannya tidak didasarkan pada *platform* dimana HTML5 itu sendiri berjalan. Yang dibutuhkan untuk memanfaatkan fitur-fiturnya hanyalah sebuah peramban *web* yang modern. Secara umum, semua peramban *web* telah mengadopsi HTML4 sebagai standar. Namun, HTML5 mulai banyak diaplikasikan pada banyak *website* di internet. Hal itu diikuti oleh peramban *web* ternama seperti Chrome, Opera, Mozilla Firefox, Safari, dan Internet Explorer yang memberikan dukungan untuk HTML5 (Sanders, 2011). Hal ini menunjukkan bahwa HTML5 akan segera menjadi standar sebagai bahasa *markup* untuk *web*.

Gambar 1 mengambarkan aplikasi HTML5 beserta komponen-komponennya. Bagian atas tertera aplikasi-aplikasi yang ada dalam HTML5, yakni HTML, CSS, dan Javascript, dimana aplikasi-aplikasi ini dibangun berdasarkan komponen-komponen yang ada di bagian bawah, yakni JavaScript API, HTML5 *markup*, dan CSS3 (Gustafson, 2013).

Ada dua organisasi yang membuat spesifikasi HTML5, yakni W3C (*World Wide Web Consortium*) dan WHATWG (*Web Hypertext Application Technology Working Group*). HTML5 pada mulanya ditulis oleh WHATWG, yang didirikan pada tahun 2003, dengan anggota-anggotanya yang terdiri dari: Apple, Mozilla, dan Opera. Pada tahun 2006, W3C membentuk sendiri kelompok kerja untuk HTML5 berdasarkan atas kerja yang telah dilakukan WHATWG. Kedua lembaga ini menghasilkan spesifikasi yang tidak terlalu jauh berbeda (Robbins, 2013).



Gambar 1. Aplikasi dan Komponen HTML5

2.1.1. HTML5 *contentEditable*

HTML5 *contentEditable* adalah sebuah fitur yang memungkinkan pengguna untuk dapat melakukan perubahan konten secara langsung pada sisi antarmuka sebuah *website*. Kode 1 menunjukkan penggunaan atribut *contentEditable* dalam sebuah struktur DOM (*Document Object Model*) yang mengubah elemen *section* menjadi area penyuntingan. Dengan memberikan nilai *true* pada atribut tersebut, maka peramban *web* akan memberikan izin untuk melakukan perubahan konten pada area yang dimaksud. Sebaliknya nilai *false*, merupakan nilai *default* pada atribut ini, menonaktifkan kemampuan untuk melakukan proses penyuntingan konten.

Kode 1. Penggunaan Atribut *contentEditable* dalam Sebuah Struktur DOM

```
1. <section contenteditable="true">
2. <h1>Edit this content</h1>
3. <p>You can select, edit, and create your own content
4. in this space</p>
5. </section>
```

2.1.2. HTML5 *Web Storage*

Fitur ini memampukan sebuah aplikasi memiliki *database* lokal yang berada di dalam peramban *web* yang digunakan, dengan kemampuan penyimpanan data yang lebih besar dibandingkan dengan kemampuan penyimpanan *cookies*. Kapasitas penyimpanan yang dimiliki oleh *web storage* bisa berbeda-beda, tergantung pada peramban *web* yang digunakan.

Keuntungan yang didapat dengan kehadiran *database* lokal ini yaitu memungkinkan aplikasi berbasis *web* untuk menyimpan data secara lokal dan dapat memanipulasi data tersebut tanpa harus terhubung dengan internet. Fitur ini telah banyak diimplementasikan oleh beberapa *website* seperti Google's Gmail, Calendar, dan Docs. HTML5 *web storage* dibagi menjadi dua macam yaitu *localStorage* dan *sessionStorage*. Data yang disimpan pada *sessionStorage* memiliki masa waktu dan akan terhapus bersamaan dengan berakhirnya *session* dari pengguna. Sedangkan data yang disimpan dalam *localStorage* akan tetap tersimpan sampai data tersebut dihapus atau diganti dengan data yang lebih baru.

2.2. *Framework*

Dalam pengembangan sebuah *website*, dibutuhkan aktivitas pemrograman yang banyak melibatkan penggunaan kode-kode maupun operasi yang berulang-ulang. Pada umumnya, penataan kode pemrograman untuk manipulasi data, penggunaan logika, dan tampilan menjadi satu dalam sebuah halaman atau berkas. Hal ini cukup merepotkan ketika pengembang harus melakukan perubahan pada program, dan lebih merepotkan lagi apabila pengembang terdiri dari beberapa orang yang tergabung dalam sebuah tim kerja. Tak jarang, hal ini disadari saat kode program telah berkembang menjadi begitu kompleks. Oleh karena itu, diperlukan sebuah kerangka kerja yang dapat menyederhanakan pengkodean program dan meminimalkan pengulangan kode-kode yang tidak diperlukan. Kerangka kerja inilah yang disebut dengan *framework*.

Secara umum, *framework* dapat diartikan sebagai alat yang dapat digunakan untuk membantu seseorang dalam melakukan sebuah pekerjaan. Definisi *framework* secara khusus adalah sebuah susunan atau rangkaian kerja yang tetap dan dibuat sedemikian rupa yang kemudian dapat digunakan kembali dalam sebuah aktifitas kerja yang lain namun tetap dalam satu area kerja yang sama dengan rangkaian kerja sebelumnya. *Web framework* memiliki definisi kumpulan sebuah maupun banyak modul-modul dalam bentuk *class library* yang dapat digunakan kembali untuk membentuk sebuah aplikasi *web* yang lebih besar dengan memanfaatkan modul-modul *class library* tersebut.

Dalam penggunaan *framework*, masih dibutuhkan penulisan-penulisan kode pemrograman yang disesuaikan dengan lingkungan *framework* yang digunakan. *Framework* menyediakan lingkungan pengembangan yang terstruktur, serta menyediakan berbagai macam fungsi siap pakai yang dapat digunakan dalam pengembangan *website*. Selain itu, tersedianya

kerangka pola kerja terstandar akan mempermudah pengembang *website* untuk memahami kode pemrograman yang ditulis oleh rekan lain saat bekerja di dalam sebuah tim kerja.

Website yang dinamis, seperti halnya *software* aplikasi, terdiri dari tiga buah komponen: *data*, *presentation*, dan *logic*. Dalam istilah *framework web*, komponen ini dikenal dengan istilah: *model*, *view*, dan *controller*. Ketika membuat *website* yang sederhana, ketiga komponen ini bercampur dan tidak menjadi masalah. Tetapi ketika membuat *website* besar dan kompleks, bercampurnya ketiga komponen ini akan menjadi masalah karena akan menimbulkan kesulitan dalam penanganannya. Untuk menangani masalah ini, ketiga komponen tersebut harus dipisahkan dalam bagian-bagian yang berbeda, dan penulisan kode program juga harus bersifat terpisah (*loosely*). Dengan menghilangkan saling ketergantungan antar komponen maka *website* yang dibuat akan lebih mudah dalam penanganannya (*manageable*), pengujian (*testable*), dan pengembangan (*scalable*) (Gilmore, 2011).

Ada banyak *framework* yang telah beredar, beberapa diantaranya yang cukup populer adalah CodeIgniter, YII Framework, Kohana, Ruby on Rails, dan CakePHP. Tentunya masing-masing *framework* memiliki standar pola kerja yang berbeda-beda, dengan masing-masing kelebihan dan kekurangannya.

2.3. FuelPHP

FuelPHP merupakan sebuah PHP *web framework* yang masih tergolong baru, sederhana, fleksibel serta didukung oleh komunitas yang dikembangkan berdasar ide-ide terbaik dari PHP *framework* lainnya. Pengembangannya dimulai sejak tahun 2010 oleh Dan Horrigan yang tidak lama kemudian berkembang dengan masuknya Phil Sturgeon, Jelmer Schreuder, dan Harro Verton ke dalam tim pengembang. Tim tersebut telah memiliki pengalaman mengenai PHP selama kurang lebih satu dekade, dan telah terlibat dalam beberapa proyek *Open-Source* seperti CodeIgniter, PyroCMS, ExiteCMS, dan DataMapper ORM.

Filosofi dari FuelPHP yaitu mengambil pendekatan yang berbeda dengan kebanyakan *framework* dan berusaha untuk menjadi *framework* yang didukung oleh komunitas. FuelPHP menjadi sebuah MVC (*Model-View-Controller*) *framework* yang didesain dengan baik untuk memiliki dukungan penuh untuk HMVC (*Hierarchial Model-View-Controller*) sebagai bagian dari arsitekturnya. Tidak hanya itu, FuelPHP juga menambahkan *ViewModel* yang merupakan sebuah *class* yang berisikan logika yang membantu dalam membangun *View* itu sendiri. Selain itu, FuelPHP juga memberikan perhatian terhadap keamanan terhadap *website* untuk mencegah penyerangan-penyerangan yang mungkin dilakukan oleh pihak-pihak yang tidak diketahui seperti XSS *attack*, SQL *injection*, dan lain sebagainya. Fitur lain yang didukung oleh FuelPHP: (1) *Class* yang bersifat modular dan *extendsible*. (2) *Oil*, sebagai utilitas baris perintah yang dapat mempercepat pengembangan *website* dengan kemampuannya yang dapat membantu dalam hal pengujian maupun pencarian kesalahan dalam pengkodean. (3) *Object Relational Mapper* (ORM) dan *Active Record* yang membantu dalam melakukan operasi pada basis data.

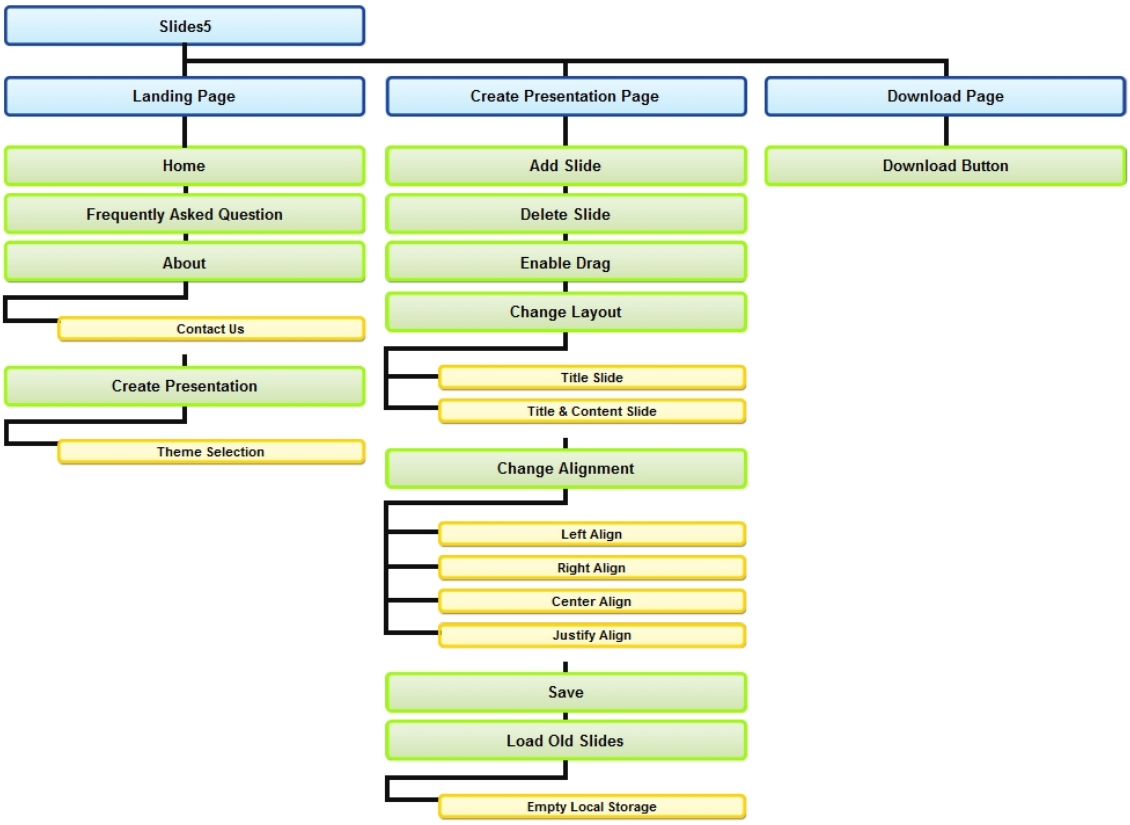
3. Analisis dan Perancangan Sistem

3.1. Desain Sistem

Gambar 2 adalah gambaran *sitemap* dari aplikasi pembuatan presentasi berbasis *web*. Pada tahapan awal, pengguna mengunjungi *website* yang dimaksud. Pada halaman utama, pengguna diberikan pilihan secara langsung untuk dapat memulai aktivitas pembuatan presentasi. Proses pertama yang dilalui adalah pemilihan tema presentasi yang akan dibuat. Sistem memberikan beberapa pilihan tema untuk dapat dipilih oleh pengguna.

Proses berikutnya adalah aktivitas pembuatan presentasi itu sendiri. Pada tahapan awal, sistem akan mengecek apakah pengguna memiliki data presentasi lama yang pernah dibuat sebelumnya. Jika pengguna memiliki data tersebut, maka akan diberikan pilihan untuk dapat memuat kembali data presentasi yang lama. Mengenai proses penyimpanan data presentasi akan dijelaskan pada bagian selanjutnya. Di halaman pembuatan presentasi, pengguna akan dibantu oleh *editor* untuk dapat melakukan pemformatan pada teks yang menjadi masukan. Selain itu,

disediakan juga beberapa fungsi untuk memanipulasi *slide* seperti menambah *slide* presentasi, menghapus *slide* presentasi, dan lain sebagainya.



Gambar 2. *Sitemap* Slides5

Setelah pengguna selesai dalam pembuatan presentasi yang diinginkan, maka pengguna dapat menyimpan presentasi yang dibuat dengan melakukan klik pada tombol *save* (*process*). Dengan mengakses tombol *save*, maka sistem akan memberikan kotak dialog untuk dapat memasukkan judul dari presentasi. Judul presentasi yang dimasukkan oleh pengguna nantinya akan menjadi *title* dari halaman presentasi yang dimaksud. Apabila judul dari presentasi telah dimasukkan, berikutnya sistem akan melakukan dua tahapan. Pertama, sistem akan mencatat isi presentasi dan disimpan dalam *localStorage* apabila peramban *web* yang digunakan oleh pengguna mendukung penggunaan *localStorage*. Kedua, sistem akan melakukan proses pencatatan isi dari presentasi yang dibuat oleh pengguna.

Setelah proses pencatatan selesai, maka presentasi telah siap untuk diunduh dan sistem akan melakukan *redirect* ke halaman unduh. Pengguna juga dapat mengunduh presentasi yang telah selesai diproses sehingga praktis untuk dibawa melalui alat penyimpanan data portabel seperti *thumbdrive*, *compact-disc*, dan lain sebagainya.

3.2. Desain Program

3.2.1. Pemilihan Tema Presentasi

Pada tahapan awal, pengguna diharuskan memilih tema yang akan digunakan untuk presentasi yang akan dibuat. Setelah pengguna memilih tema, maka sistem akan melakukan proses berikutnya untuk menciptakan presentasi baru.

3.2.2. Pembuatan Presentasi

Proses berikutnya adalah proses pembuatan presentasi itu sendiri. Pada halaman pembuatan presentasi, pengguna akan dibantu oleh beberapa fungsi JavaScript pendukung

pembuatan presentasi. Pemanfaatan fungsi dalam bentuk JavaScript yang bersifat *client-side* akan sangat membantu dalam proses pembuatan presentasi, sehingga meminimalkan lalu lintas antara pengguna dan *server*. Beberapa fungsi pendukung itu antara lain: *add slide, delete slide*, *change layout*, *change alignment*, *enable dragging*, *save*, *download*, *load old slide.*

3.3. Implementasi dan Pengujian

Pada tahapan awal, akan dilakukan pembuatan program per bagian kecil. Seperti yang telah dijelaskan sebelumnya, FuelPHP akan digunakan sebagai *framework* dalam pengembangan aplikasi presentasi berbasis *web*. Dengan pengembangan per bagian kecil, diharapkan dapat mempercepat pengembangan dengan berfokus pada masing-masing bagian tersebut. Selain melakukan pengembangan, juga dilakukan pengujian untuk masing-masing komponen yang telah selesai dibuat. Hal ini dilakukan untuk menguji apakah fungsi dari komponen tersebut sudah dapat berjalan dengan baik dan benar.

Tahapan berikutnya yaitu melakukan proses integrasi untuk masing-masing komponen yang telah dibuat dan diuji sebelumnya. Di samping itu, pengujian juga dilakukan terhadap aplikasi secara keseluruhan untuk menguji kembali fungsi dari masing-masing komponen. Pengujian juga mencakup bagaimana interaksi antara pengguna dan sistem. Semuanya dilakukan untuk mendapatkan hasil yang maksimal dengan meminimalkan kesalahan-kesalahan maupun hal-hal yang kurang berguna.

Setelah penggabungan dan pengecekan aplikasi secara utuh telah selesai, maka akan dilakukan optimisasi dengan menggunakan Smart Optimizer. Seperti yang telah dijelaskan sebelumnya, Smart Optimizer berguna dalam melakukan kompresi berkas-berkas *web* sebelum dikirim kepada pengguna agar beban pengunduhan halaman dapat dioptimalkan. Hal ini sangat berguna khususnya di halaman pembuatan presentasi karena ada cukup banyak berkas *web* yang harus diunduh dalam sekali waktu.

3.4. Operasi dan Pemeliharaan

Tahapan terakhir yaitu tahap pengoperasian dan pemeliharaan terhadap aplikasi berbasis *web* yang dibuat. Pengoperasian dilakukan dengan melakukan mengunggah berkas-berkas *website* dari posisi lokal ke *server web*. Di samping itu, pemeliharaan juga dilakukan untuk mengetahui kinerja sistem secara utuh.

Salah satu pemeliharaan yang dilakukan dalam *website* ini adalah penghapusan berkas-berkas presentasi yang sudah tidak terpakai. Setiap kali pengguna membuat presentasi, akan tercipta satu berkas presentasi baru di *server*. Hal ini akan menyebabkan penumpukan berkas-berkas presentasi seiring dengan semakin banyaknya pengguna yang memanfaatkan *website* ini. Oleh karena itu dibutuhkan sebuah fungsi yang dapat mengatur penghapusan berkas-berkas presentasi dengan batas tertentu. *Website* akan menyimpan semua file presentasi yang pernah dibuat dalam kurun waktu 30 hari. Berkas presentasi yang memiliki usia lebih dari 30 hari terhitung sejak tanggal pembuatan akan dihapus dari *server*.

Fungsi penghapusan berkas ini nantinya akan di*set* dengan menggunakan *cron job* sehingga akan berjalan dengan sendirinya tanpa perlu secara manual mengakses fungsi tersebut. Selain itu, tidak menutup kemungkinan untuk dilakukan pengembangan ke arah yang lebih lanjut untuk membuat aplikasi terkait dapat berguna serta berfungsi dengan lebih baik lagi.

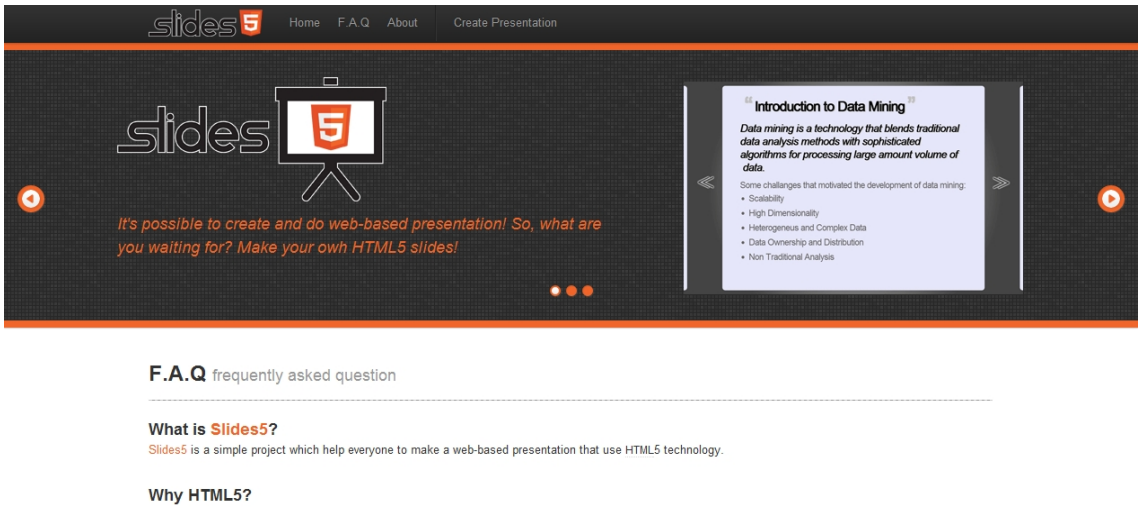
4. Hasil dan Pembahasan

Pada bagian ini, akan dijelaskan hasil dari aplikasi yang telah dibuat beserta dengan beberapa gambar untuk mendukung penjelasan yang ada. Pengujian dapat dilakukan langsung dengan mengakses alamat http://slides5.com/.

4.1. *Front End*

Front end dari *website* Slides5 (Gambar 3) menerapkan prinsip *single page website*. *Single page website* menjadi sebuah cara penyajian yang baik dengan mengumpulkan semua informasi yang berguna mengenai *website* terkait dalam satu halaman utuh. Hal ini tentunya

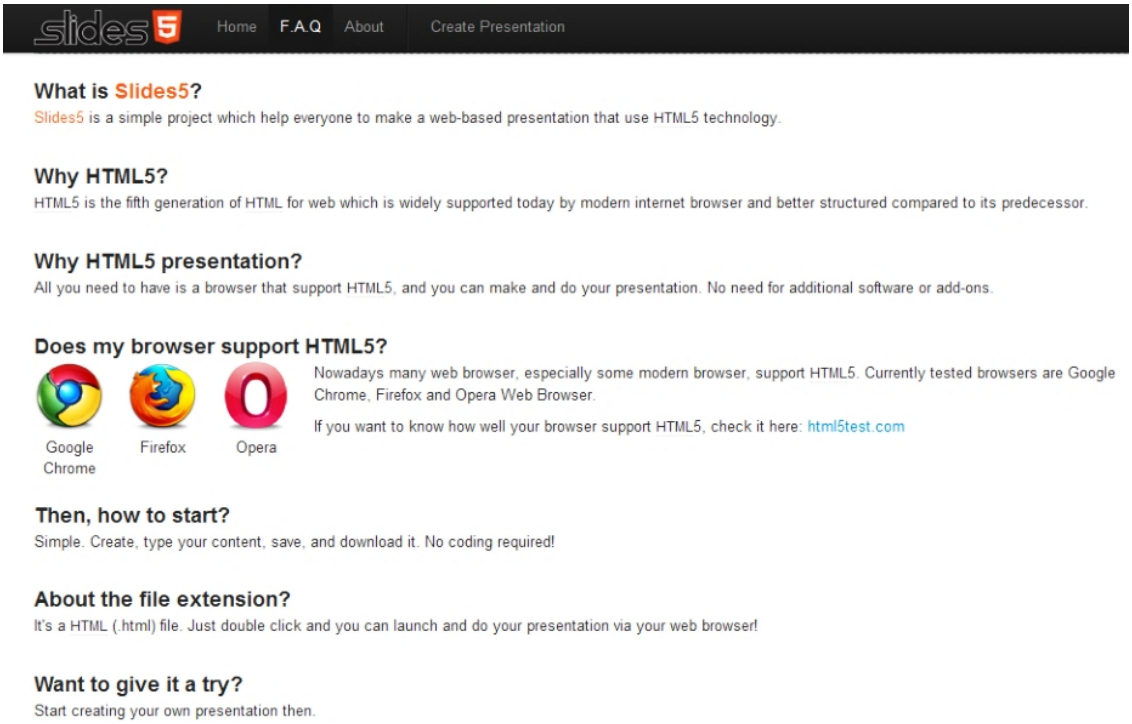
memberikan keuntungan tersendiri bagi pengunjung yang mengakses *website* tersebut karena semua kode yang dibutuhkan seperti HTML, JavaScript, dan CSS didapatkan dengan sekali unduh. Di samping itu, hal ini juga memudahkan pemrogram dalam melakukan pengembangan *website* sehingga tidak perlu menciptakan halaman *view* yang terlalu banyak. Setelah semua komponen yang dibutuhkan telah siap, maka pengguna akan mendapatkan akses ke semua informasi dasar yang ada di *website* terkait. Halaman utama dari Slides5 dibagi menjadi tiga bagian kecil yaitu *frequently asked question*, *about,* dan *create presentation*.



Gambar 3. Halaman Beranda Slides5

4.1.1. *Frequently Asked Question*

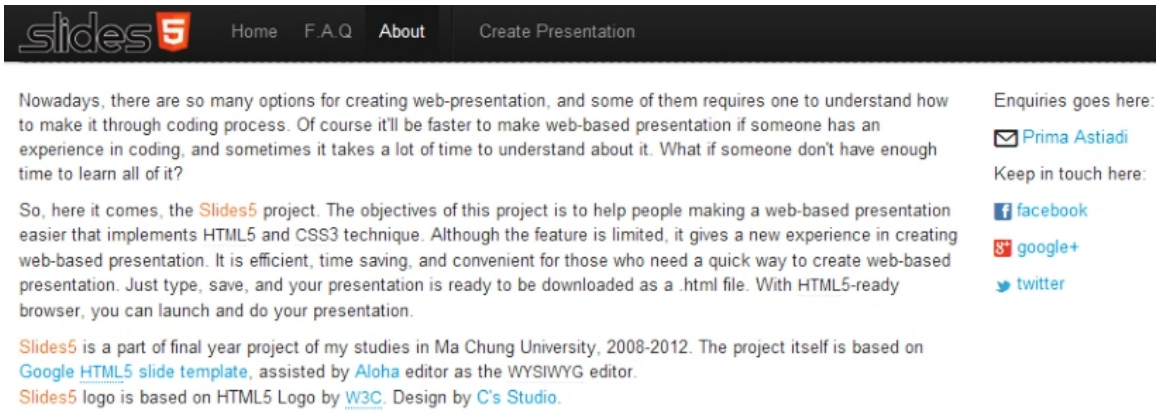
Bagian ini (Gambar 4) berisi tentang segala informasi dasar yang mungkin dibutuhkan oleh pengguna saat pertama kali mengunjungi *website* ini. Selain itu bagian ini juga berisi informasi yang harus diperhatikan oleh pengguna seperti informasi mengenai peramban *web* yang didukung, ekstensi dari berkas file yang diciptakan, dan lain sebagainya.



Gambar 4. Bagian *Frequently Asked Question* pada Slides5

4.1.2. *About*

Bagian ini (Gambar 5) berisi penjelasan singkat seputar Slides5 seperti tujuan, dan kegunaan dari *website* terkait. Selain itu bagian ini juga berisi informasi berupa alamat *email* yang dapat dituju apabila pengguna memiliki pertanyaan mengenai *website* terkait.

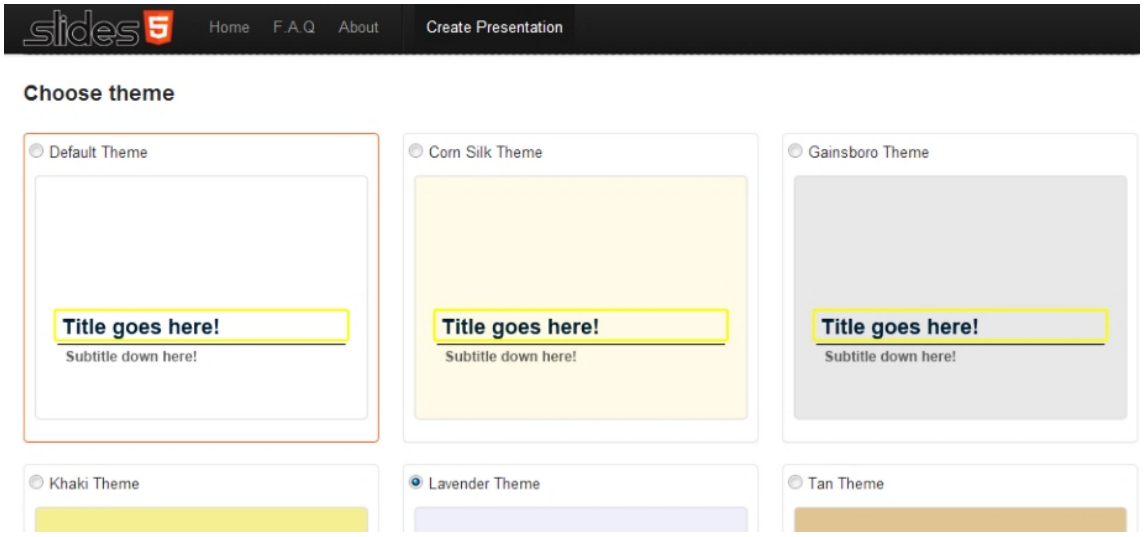


Gambar 5. Bagian *About* pada Slides5

4.1.3. *Create Presentation*

Bagian ini (Gambar 6) adalah bagian dimana pengguna dapat memulai untuk membuat presentasi. Pada bagian ini, pengguna dihadapkan dengan beberapa pilihan tema yang telah disediakan. Saat *website* ini dikembangkan, terdapat sembilan buah pilihan tema dan tidak menutup kemungkinan untuk tambahan tema-tema lain di masa mendatang.

Untuk memulai membuat presentasi, pengguna harus memilih salah satu tema yang ada. Setelah pengguna selesai memilih tema yang diinginkan, pengguna dapat melanjutkan ke tahapan berikutnya dengan mengakses tombol *proceed* yang telah disediakan.

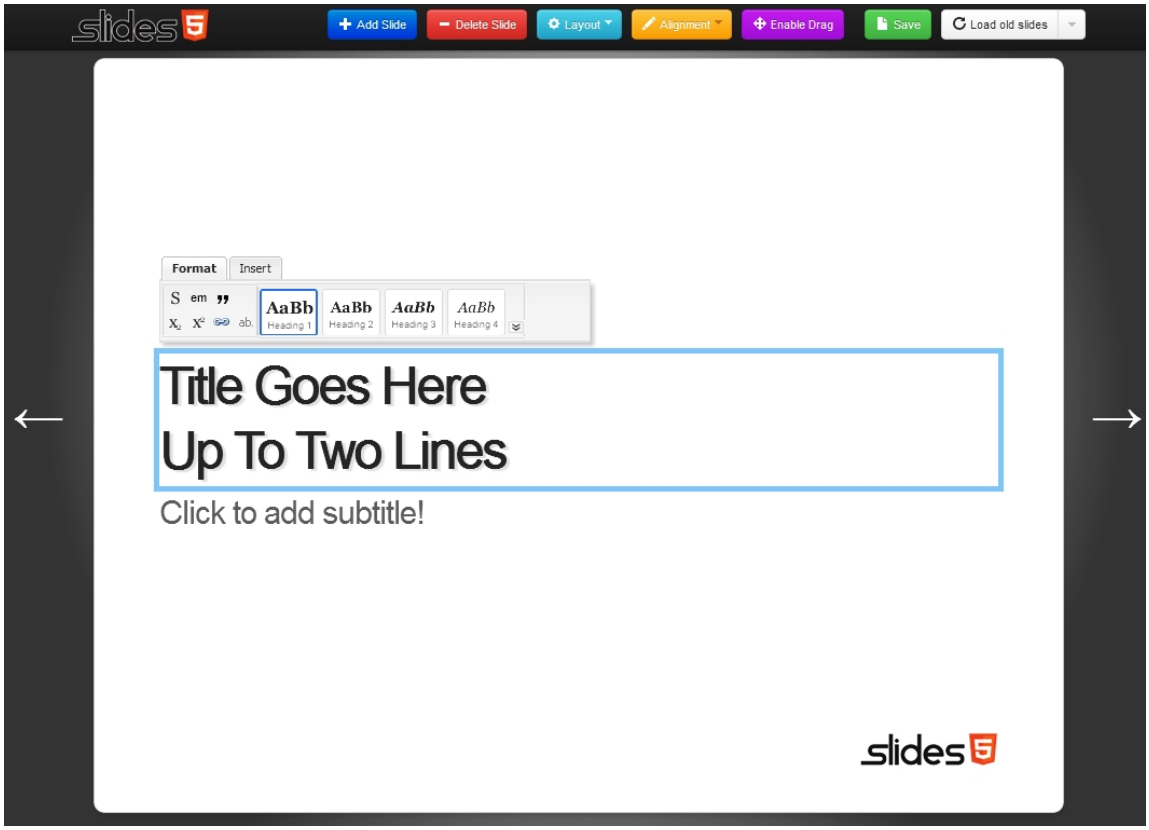


Gambar 6. Bagian *Create Presentation* pada Slides5

4.2. Halaman Pembuatan Presentasi

Halaman ini (Gambar 7) adalah halaman dimana pengguna dapat mulai berinteraksi dengan aplikasi untuk menciptakan presentasi yang diinginkan. Pada halaman ini, pengguna akan bekerja dengan tema presentasi yang telah dipilih pada proses sebelumnya. Berikut adalah tampilan dari halaman pembuatan presentasi.

Seperti yang terlihat pada Gambar 7, halaman pembuatan presentasi ini dapat dibagi menjadi tiga bagian utama, yaitu: (1) *Action panel* (2) Area penyuntingan (3) Area navigasi.



Gambar 7. Halaman pembuatan presentasi

4.2.1. *Action Panel*

Bagian ini terletak di sisi atas halaman pembuatan presentasi, berisi fungsi-fungsi yang berguna dalam membantu pengguna untuk membuat presentasi. Semua fungsi yang ada memiliki kegunaan masing-masing yang nantinya dapat dilihat pada area penyuntingan.

4.2.2. Area Penyuntingan

Bagian ini dapat dikatakan sebagai bagian utama dalam pembuatan presentasi. Dibantu dengan Aloha Editor, pengguna dapat memasukkan konten presentasi yang diinginkan. *Layout* dari area penyuntingan ini dapat dimanipulasi dengan fungsi-fungsi yang ada pada panel aksi.

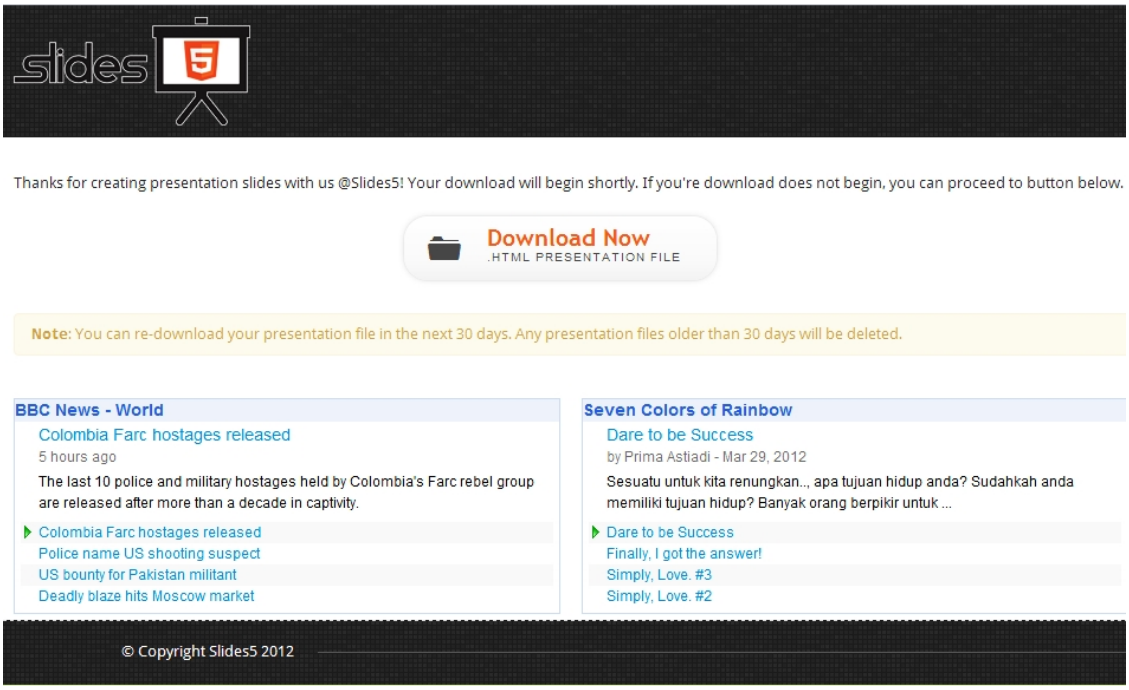
4.2.3. AreaNavigasi

Bagian ini adalah bagian dimana pengguna dapat melakukan navigasi pada area penyuntingan, seperti berpindah maju ke *slide* berikut atau mundur ke *slide* sebelumnya. Ada dua macam cara untuk dapat melakukan navigasi, yaitu dengan melakukan klik pada area navigasi atau dengan menggunakan *keyboard*. Untuk navigasi dengan menggunakan *keyboard*, beberapa tombol yang dapat dipakai adalah sebagai berikut: (1) *Next slide*: dapat diakses dengan menggunakan tombol spasi, tombol *page down*, tombol anak panah kanan, atau tombol anak panah bawah pada *keyboard.* (2) *Previous slide*: dapat diakses dengan menggunakan tombol anak panah kiri, tombol *page up*, tombol *backspace*, atau tombol anak panah atas yang ada di *keyboard*.

4.3. Halaman Unduh

Dari halaman pembuatan presentasi, pengguna akan diarahkan ke halaman ini untuk dapat mengunduh berkas presentasi yang telah dibuat sebelumnya dalam bentuk berkas HTML. Gambar 8 adalah tampilan dari halaman unduh. Proses pengunduhan diatur supaya setelah 5 detik, kotak dialog untuk mengunduh berkas akan muncul dan pengguna dapat mulai menentukan lokasi penyimpanan berkas presentasinya. Apabila ada suatu permasalahan

sehingga kotak dialog untuk mengunduh berkas tidak muncul dalam waktu 5 detik, maka pengguna dapat melakukan *request* kembali dengan melakukan klik pada tombol unduh yang juga disediakan di halaman tersebut.



Gambar 8. Halaman unduh pada Slides5

5. Kesimpulan dan Saran

5.1. Kesimpulan

Melalui serangkaian uji coba terhadap layanan yang telah dikembangkan, didapatkan hasil sebagai berikut: (1) Slides5 sebagai layanan pembuatan presentasi berbasis *web* dapat membantu dalam membuat materi presentasi berbasis *web* dengan mudah. (2) Berkas presentasi dalam format HTML yang dapat diterima oleh mayoritas peramban *web* dapat menjadi solusi atas permasalahan kompatibilitas yang ditemui pada aplikasi *standalone* pembuatan presentasi.

5.2. Saran

(1) Penggunaan Aloha Editor sebagai *editor* pembantu cukup membebani sistem, ditunjukkan dengan lamanya *page loading time* yang mencapai tiga menit dengan kecepatan internet sebesar 30 kbps. Diharapkan dalam proses pengembangan berikutnya dapat menggunakan *editor* lain dengan performa yang lebih baik. (2) Slides5 saat ini hanya berfokus untuk menerima masukan berupa teks saja. Diharapkan dalam pengembangan berikutnya dapat ditambahkan fitur untuk menerima masukan lain seperti data gambar atau yang lainnya.

Referensi

Hogan, B.P. 2013. *HTML5 and CSS3: Level Up with Today's Web Technologies, Second Editon*. Dallas, TX, USA: PragmaticProgrammers.

Clark, R., Studholme, O., Murphy, C., & Manian, D. 2012. *Beginning HTML5 and CSS 3*. New York, NY, USA: Apress.

David, M. 2010. *HTML5 Designing Rich Internet Application*. Burlington, MA, USA: Elsevier.

Devlin, I. 2012. *HTML5 Multimedia: Develop and Design*. Berkeley, CA, USA: Peachpit.

Gilmore, W.J. 2011. *Easy PHP Websites with the Zend Framework*. USA: WJ Gilmore.

Goldstein, A., Lazaris, L., & Weyl, E. 2011. *HTML5 & CSS3 for the Real World.* Collingwood, VIC, Australia: SitePoint.

Gustafson, J.M. 2013. *HTML5 Web Application Development By Example: Beginner's guide*. Birmingham, UK: Packt.

Powell, T.A. 2010. *HTML & CSS: The Complete Reference, Fifth Edition*. New York, NY, USA: McGraw-Hill.

Robbins, J.N. 2013. *HTML5 Pocket Reference, Fifth Edition*. Sebastopol, CA, USA: O'Reilly.

Sanders, B. 2011. *Smashing HTML5, FirstEdition*. Chichester, West Sussex, UK: John Wiley & Sons.