
Pengembangan *Knowledge Management System* Daring untuk Memproses dan Mengekstrak Pengetahuan dalam Dokumen

Krishna Aji Satriatama*¹, Ibnu Santoso²

¹IVKS3/16.9227

e-mail: *16.9227@stis.ac.id, ibnu@stis.ac.id

Abstrak

Informasi dalam dokumen seringkali terlupakan dan hilang apabila tidak disimpan dengan baik. Politeknik Statistika STIS adalah salah satu institusi yang memerlukan akses mudah kepada informasi dan dokumen di dalamnya, terutama mahasiswa. Mahasiswa Politeknik Statistika setiap tahun mengadakan penjualan modul kumpulan soal-soal UTS dan UAS tahun sebelumnya kepada mahasiswa setiap anggota angkataannya dalam bentuk cetak. Namun dokumen untuk soal ini dipegang oleh perorangan dan tidak semua mahasiswa mempunyai akses kepada dokumen tersebut selain dari modul tadi. Penelitian ini bertujuan untuk mencari, mendesain, dan mengimplementasikan sebuah aplikasi manajemen pengetahuan yang membantu mahasiswa untuk menyimpan dan mendistribusikan informasi berbasis web. Web ini dikembangkan menggunakan kerangka pengembangan “Laravel” dan di dalamnya terdapat fitur untuk mengekstrak teks dari sebuah dokumen atau hasil pemindaian dokumen. Berdasarkan pengujian yang dilakukan menggunakan metode White Box dan Black Box testing, aplikasi sudah dapat menjalankan fungsinya dengan baik. Selain itu, System Usability Scale juga memberikan hasil yang baik terhadap pengalaman pengguna di dalam aplikasi. Dengan dikembangkannya aplikasi ini, maka diharapkan dapat mempermudah setiap mahasiswa dalam mengakses, menyimpan, dan mendistribusikan dokumen.

Kata kunci—Knowledge management, web development, text mining, Badan Pusat Statistik

Abstract

Information from document is often lost if not stored properly. Polytechnic of Statistics STIS is one of many institutions that need access to that information with ease, especially their students. Students here annually release printed compilation of documents from previous exams, organized by students from each year. But these documents are held by individuals, and those who doesn't have privilege cannot access them other than that releases. This study will find, explore, design, and implement a knowledge management application that helps students to store and share documents inside website. This website is developed using “Laravel” framework that also supports extracting texts from scanned physical document. Based on result of White Box and Black Box tests, this application can perform all its functions well. System Usability Scale survey also give “Good” score which resembles user's experience within application. With this application developed, hopefully each student will have easy access to share and distribute documents.

Keywords— Knowledge management, web development, text mining, Badan Pusat Statistik

1. PENDAHULUAN

Pengetahuan atau *knowledge* adalah suatu aset yang sangat berharga dari sebuah organisasi. Semenjak tahun 1990-an, *Knowledge Management* (KM) sudah menjadi permasalahan bagi suatu organisasi (Mohajan, 2016). Teknologi yang terus berkembang pesat seiring dengan berkembangnya ilmu pengetahuan membuat organisasi menjadi sadar akan pentingnya *Knowledge Management System* (KMS) yang mampu menyimpan pengetahuan agar dapat menjamin keutuhan dan keberadaan pengetahuan tersebut. Politeknik Statistika STIS adalah merupakan institusi yang dibawah oleh Badan Pusat Statistik (BPS) dan organisasi yang sangat mengapresiasi adanya KMS ini. *Knowledge management* ini sendiri merujuk ke sebuah sistem yang terstruktur dengan tujuan utama untuk meningkatkan kinerja dari sebuah organisasi dalam pengambilan, penyimpanan, dan pendistribusian informasi (Nevis *et al.*, 1995).

Di Politeknik Statistika STIS, mahasiswa dan dosen masih menggunakan aplikasi “*Google Drive*” yang merupakan aplikasi sangat baik untuk penyimpanan dan pembagian fail atau dokumen. Namun aplikasi tersebut bersifat pribadi dan kontennya tidak bias diakses oleh publik dengan mudah. Jika satu mahasiswa tidak menanyakan akses kepada pemilik dokumen, maka tidak ada cara lain bagi mahasiswa tersebut untuk mendapatkan dokumennya. Ada pula kasus dimana mahasiswa ingin mempersiapkan untuk Ujian Tengah Semester (UTS) dan Ujian Akhir Semester (UAS), maka angkatan tiap tahun akan membuat kompilasi soal UTS/UAS tahun sebelumnya dan dicetak dalam bentuk buku teks untuk dijual kepada yang membutuhkan. Idealnya, buku teks itu seharusnya dibagikan kepada seluruh mahasiswa agar semuanya mempunyai kesempatan sama untuk mempersiapkan ujian mereka. Namun, tidak semuanya dapat membeli dan mencetak buku akan memakan biaya waktu, sumber daya, dan dana. Selain hal itu, dokumen kompilasi tersebut hanya dipegang oleh perorangan yang terikat dalam organisasi angkatan yang berarti jika ada yang membutuhkan dan orang tadi sudah keluar dari organisasi berkemungkinan untuk hilang. Maka diperlukan sebuah tempat untuk menyimpan dan membagikan dokumen yang mudah diakses oleh seluruh mahasiswa dimana disana mahasiswa bisa mencari apa yang mereka mau dan ikut berkontribusi untuk mengembangkan tempat itu menjadi lebih lengkap.

Karena membuat konten *knowledge base* tidaklah sama seperti membuat konten *blog* pada biasanya, maka diperlukan desain yang sedemikian rupa menarik orang untuk datang dan berkontribusi untuk mengisi artikel di dalamnya. Namun hal ini tidak mudah, karena saling berbagi pengetahuan hanya akan terjadi apabila kontributor dan pencari pengetahuan itu ada (Bock *et al.*, 2006).

Untuk mewujudkan hal tersebut maka diperlukan sebuah aplikasi yang bisa mengatasi permasalahan seperti yang terjadi di Politeknik Statistika tadi dengan membuat sebuah sistem yang mampu mengunggah sebuah dokumen, memproses dokumen tersebut dengan metode ekstraksi dokumen, dan menampilkannya menjadi sebuah artikel yang menjelaskan isi dokumen tersebut. Hal ini dilakukan untuk meminimalisasi jumlah kata yang kontributor tulis. Selain itu sistem juga memiliki fungsi pencarian cepat untuk mendapatkan apa yang dicari untuk selanjutnya bisa dilihat, diunduh, atau disunting apabila konten dinilai tidak sesuai. Pembangunan aplikasi akan dilakukan menggunakan *framework* “*Laravel*” yang dipasang dalam paket *server* “*XAMPP*”.

2. METODOLOGI

Berisi landasan teori, cakupan penelitian (termasuk sumber data dan metode pengumpulan data), serta metode analisis yang digunakan.

2.1 Landasan Teori

2.1.1 Knowledge Management System (KMS)

KMS adalah sebuah sistem yang digunakan untuk membantu membuat, membagi, mengatur, dan memakai informasi dan pengetahuan dalam sebuah organisasi dengan tujuan utama untuk meningkatkan performa kerja organisasi secara umum (O'Dell & Grayson, 1998). Selain itu KMS juga dapat diaplikasikan dalam beberapa teknologi seperti: *expert system*, *cognitive science*, *library science*, *technical writing*, *document management*, *decision support system*, *semantic networks*, *object-oriented information modelling*, *full-text search & retrieval*, dan *performance support system* (University of North Carolina, 2007). Pengaplikasian KMS pada penelitian ini merujuk kepada *document management* untuk fokus kepada penyimpanan dan pembagian dokumen kepada anggota organisasi.

2.1.2 Software Development Cycle (SDLC)

SDLC adalah sebuah pendekatan yang *developer* dan *programmer* gunakan untuk membangun sebuah proyek sistem informasi yang terstruktur mulai dari pencarian kebutuhan sistem, studi kelayakan sistem hingga rilis aplikasi tersebut. Ada beberapa jenis metode SDLC ini, seperti *waterfall*, *rapid application development*, *spiral*, dan *unified* (Ruparelia, N. B., 2010). Metode yang dipakai pada penelitian ini adalah metode *waterfall*. Metode ini merupakan metode yang berurutan dan terstruktur dalam prosesnya yang digambarkan turun temurun layaknya sebuah air terjun dan proses setiap terjunnya harus dilakukan secara berurutan untuk menghasilkan sebuah aplikasi yang sukses (Bassil Y, 2012). Namun metode ini juga memiliki kekurangan ketika dilakukan peninjauan ke tahap sebelumnya dan setiap proses harus dilakukan tahap demi tahap, dimana ini kurang efisien dalam pengembangan aplikasi yang harus selalu di testing (Royce, 1970).

Berdasarkan definisi menurut Royce, tahapan dalam metode *waterfall* antara lain adalah:

1. *System Requirement*
Spesifikasi dari sistem dimana aplikasi akan berjalan. Dalam penelitian ini sistem akan berjalan di *web*, yang berarti aplikasi akan bersifat *cross-platform* atau dapat dijalankan di banyak tempat seperti komputer, telepon genggam, atau tablet.
2. *Software Requirement*
Spesifikasi dari perangkat lunak yang digunakan untuk membangun aplikasi. Dalam penelitian ini aplikasi yang dipakai adalah *Microsoft Windows*, *XAMPP stack server*, *Laravel*, *Bootstrap*, *Visual Studio Code*, *Notepad++*, *Composer package installer*, dan *Github for Windows*.
3. *Analysis*
Tahapan ini membahas dari perilaku yang aplikasi dapat lakukan. Perilaku ini di definisikan berdasarkan kebutuhan dari pengguna dan interaksi yang dilakukan antara pengguna dengan aplikasi. Hasil dari analisis ini dapat berupa tujuan penggunaan aplikasi, fungsi di dalam aplikasi, karakteristik pengguna, *database requirement*, dan *interface requirement*.
4. *Program Design*
Proses ini adalah proses dimana desainer dan *programmer* menggambarkan bagaimana hasil analisis diatas dapat diimplementasikan menjadi aplikasi dengan hasil akhir seperti desain algoritma, arsitektur aplikasi, skema *database*, desain tatap muka, dan *logical diagram*.

5. *Coding / Implementation*

Implementasi adalah proses realisasi dari hasil desain menjadi sebuah aplikasi yang satu dan berfungsi. Proses ini adalah proses dimana *programmer* akan mengkode programnya menggunakan bahasa pemrograman dan akan disusun menjadi aplikasi yang sudah dapat dijalankan oleh pengguna.

6. *Testing*

Sebelum aplikasi dapat dirilis kepada pengguna, aplikasi akan di tes menggunakan beberapa prosedur validasi dan verifikasi apakah aplikasi sudah dibuat dan dapat dijalankan sesuai dengan spesifikasi yang sudah dianalisis pada tahap sebelumnya. Contoh testing yang dilakukan adalah *White Box Testing* yang merupakan *high-level* testing menyeluruh hingga kode yang ditulis dan dilakukan oleh *programmer* yang menuliskan kode pengujiannya dan *Black Box Testing* yang merupakan *low-level* testing dilakukan oleh pengguna aplikasi untuk mengetes apakah aplikasi berfungsi seperti seharusnya (Nidhra, S. & Dondeti, J., 2012).

7. *Maintenance*

Proses ini adalah proses terakhir yang dilakukan untuk menjaga keutuhan aplikasi jika ada masalah, meningkatkan performa apabila ditemukan metode yang lebih baik, adaptasi dengan teknologi baru, atau menerapkan fungsi baru berdasarkan timbal balik pengguna.

2.1.3 XAMPP

XAMPP adalah sebuah paket *server* yang berisikan aplikasi *cross-platform* *Apache HTTP Server*, *MariaDB*, *PHP*, dan *Perl*. Aplikasi ini digunakan untuk memudahkan pengguna dalam membangun aplikasi web beserta *server* dan *database* di dalamnya. Aplikasi ini bersifat *open-source* dan berlisensikan *GNU General Public License* yang berarti gratis dan bebas untuk di distribusikan, namun tidak untuk memodifikasi paket di dalamnya (GNU.org, 2007).

2.1.4 Laravel

Laravel adalah sebuah *framework* untuk pembangunan aplikasi berbasis *web* bermodelkan *Model-View-Controller* (MVC). Aplikasi ini dibuat untuk mempermudah pembangunan *web* bagi *developer* dan mempunyai banyak fungsi di dalamnya sehingga pengguna tidak perlu menuliskan kode dari dasar. *Laravel* dipilih di penelitian ini karena dokumentasinya yang jelas, adanya *Laracast* atau tutorial pemakaian *Laravel* dalam format *video*, dan penulisan kodenya yang elegan dibandingkan *CodeIgniter* setelah dicoba penggunaannya pada sebelum *software requirement*, terutama pada fungsi *modelling* dan *controller*-nya.

2.1.5 MySQL

MySQL adalah sebuah *database* yang bersifat *open-source* yang digunakan untuk menyimpan data dan tabel di dalamnya. *MySQL* ini dipilih karena sudah termasuk kedalam paket *XAMPP* dan penggunaannya yang paling banyak disukai berdasarkan jumlah bintang pada situs *Github*.

2.2 Cakupan Penelitian

Cakupan untuk penelitian ini meliputi dari mulai pembangunan sebuah aplikasi berbasis *web* dengan fitur utama penyimpanan, pencarian, dan pengunduhan dokumen. Aplikasi ini dibangun dengan pendekatan metode *waterfall* dengan harapan pengerjaan dapat berjalan dengan sistematis.

2.3 Metode Analisis

2.3.1 Analisis Sistem Berjalan

Pada saat ini, belum ada sistem yang bisa digunakan di Politeknik Statistika STIS untuk mencari dan berbagi informasi atau dokumen yang bisa diakses oleh seluruh civitas akademika. Selama ini apabila ingin berbagi, maka harus menggunakan aplikasi “*Google Drive*” untuk menyimpan dokumen yang ingin dibagi.

2.3.2 Analisis Permasalahan

Permasalahan yang timbul dari tidak adanya sistem saling berbagi adalah apabila mahasiswa ingin mencari materi, dokumen, atau informasi akademik maka mahasiswa tersebut harus bertanya ke orang lain apabila orang tersebut memiliki apa yang dia cari. Begitu juga dengan dosen yang ingin membagikan materi atau dokumen kepada mahasiswa, maka beliau harus mengirimnya melalui surel kepada seluruh mahasiswanya atau kepada mahasiswa terpilih untuk dilanjut kepada mahasiswa lainnya.

2.3.3 Analisis Sistem Usulan

Berdasarkan permasalahan pada analisis permasalahan, maka diusulkan untuk dibangun sebuah sistem yang mampu menyimpan dan membagi informasi atau dokumen bagi seluruh mahasiswa di Politeknik Statistika STIS.

2.4 Metode Evaluasi

Pada fase evaluasi akan dilakukan testing kepada aplikasi. Testing paling sering digunakan untuk memverifikasi dan validasi kualitas dari aplikasi yang dikembangkan (Shao, D, et al, 2007). Metode yang dipakai untuk testing ini ada:

1. *Black Box & White Box Testing*

Black Box testing atau bisa disebut juga *functional testing* adalah tes yang dilakukan untuk menguji berbagai kasus untuk melihat apakah hasil implementasi aplikasi sesuai dengan *requirement* atau *design specification* (Nidhra, S., & Dondeti, J., 2012). Pengujian ini dilakukan kepada sasaran pengguna untuk selanjutnya diberikan timbal balik apakah aplikasi berjalan dengan lancar dan memenuhi tujuan awal. *Functional test* ini fokus hanya kepada perilaku yang tampak dari luar (*front-end*) aplikasi (Murnane, T., & Reed, K., 2001) dan tidak melihat bagaimana aplikasi itu bekerja dari sisi *back-end*. Sedangkan *white box* testing merupakan uji coba yang pengembang jalankan untuk mengetes apakah fungsi-fungsi dalam aplikasi dan *unit* di dalamnya bekerja sesuai dengan harapan dengan tidak menghasilkan *error* dalam prosesnya. Uji *white box* ini digunakan sebagai sarana bagi pengembang untuk mengotomatisasi pengujian yang dilakukan berkali-kali tanpa harus secara manual mencobanya.

2. *System Usability Scale (SUS)*

SUS adalah skala penilaian yang dihasilkan dari 10 pertanyaan yang diberikan kepada calon pengguna untuk melihat apakah sistem yang dikembangkan dapat dipakai dan diterima dengan skala “Sangat Tidak Setuju” sampai “Sangat Setuju”. SUS ini dipakai karena diasumsikan penilaiannya yang *unidimensional*, hanya fokus kepada bagaimana pengalaman pengguna, sehingga dapat menjelaskan kegunaan sistem dari satu nilai akhir dalam SUS (Lewis et al, 2009). Perlu diperhatikan juga bahwa nilai SUS dari perorangan sama sekali tidak bisa digunakan untuk merepresentasikan kegunaan sistem. Akurasi dari setiap survei akan meningkat untuk setiap sampel dan akurasi SUS dapat menghasilkan akurasi yang sangat baik pada ukuran 12 sampel (Tullis & Stetson, 2004).

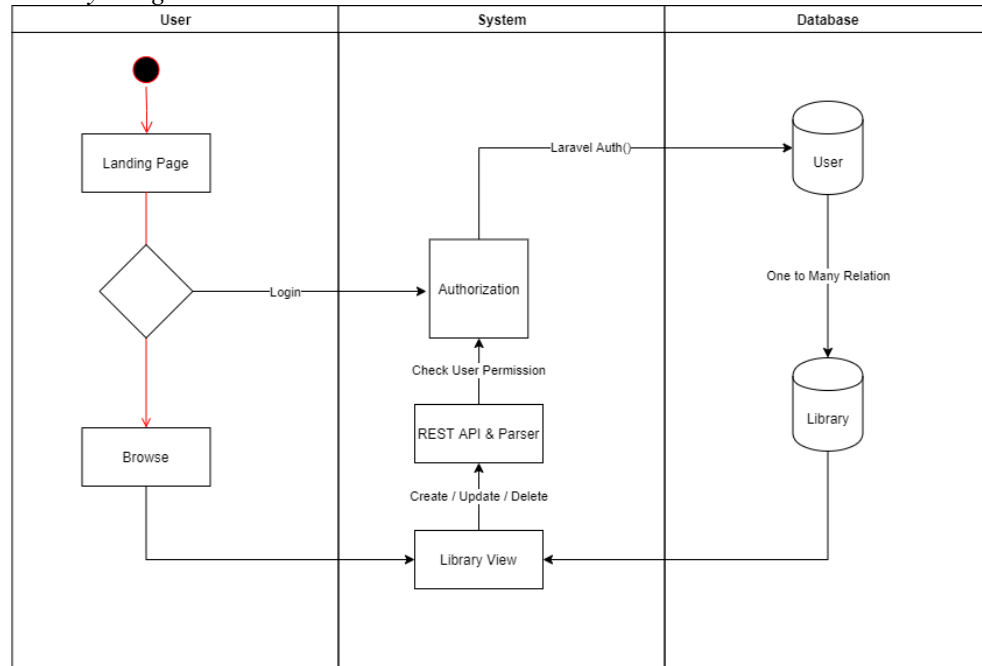
3. HASIL DAN PEMBAHASAN

Pembahasan terhadap hasil penelitian dan pengujian yang diperoleh disajikan dalam bentuk uraian teoritik, baik secara kualitatif maupun kuantitatif. Hasil olah data secara deskriptif ditampilkan dalam bentuk grafik ataupun tabel.

3.1 Rancangan Sistem Usulan

Berdasarkan analisis pada bab sebelumnya, rancangan sistem yang dibangun dapat dijelaskan dalam beberapa diagram dibawah ini:

1. Activity Diagram

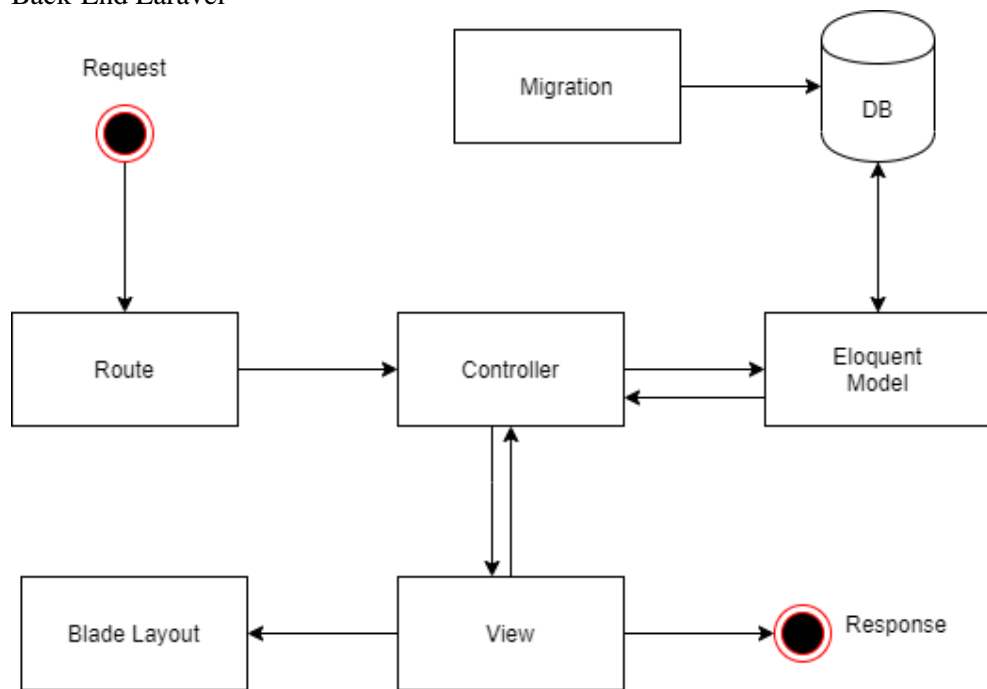


Gambar 1. Activity Diagram

Dalam diagram ini akan dijelaskan bagaimana cara kerja aplikasi dan interaksinya dengan pengguna. Deskripsi dari diagram diatas adalah:

- Pengguna masuk kedalam situs, akan pertama kali disambut oleh *landing page* yang bermuatan logo, judul situs, dan pilihan ke beranda/*browse* atau *login*.
- Pengguna dapat menjelajahi isi dari situs untuk melihat artikel di dalamnya dan mengunduh dokumen yang tersedia dalam artikel.
- Apabila pengguna ingin membuat artikel dan/atau mengunggah dokumen baru, maka pengguna diharuskan untuk *login* terlebih dahulu. Pengguna akan di otentikasi kepada *database* dan jika berhasil maka pengguna dapat melakukan pembuatan artikel baru dan pengunggahan.
- Apabila pengguna tidak mempunyai akun, maka dapat dilakukan pendaftaran yang akan disimpan dalam *database*.
- Apabila pengguna mengunggah dokumen baru, maka sistem akan mencoba *parsing* dokumen tersebut untuk menjadi konten dari artikel yang dibuat melalui API yang disediakan di dalam aplikasi.
- Semua informasi dalam artikel akan disimpan dalam *database* milik artikel tersebut.

2. Back-End Laravel

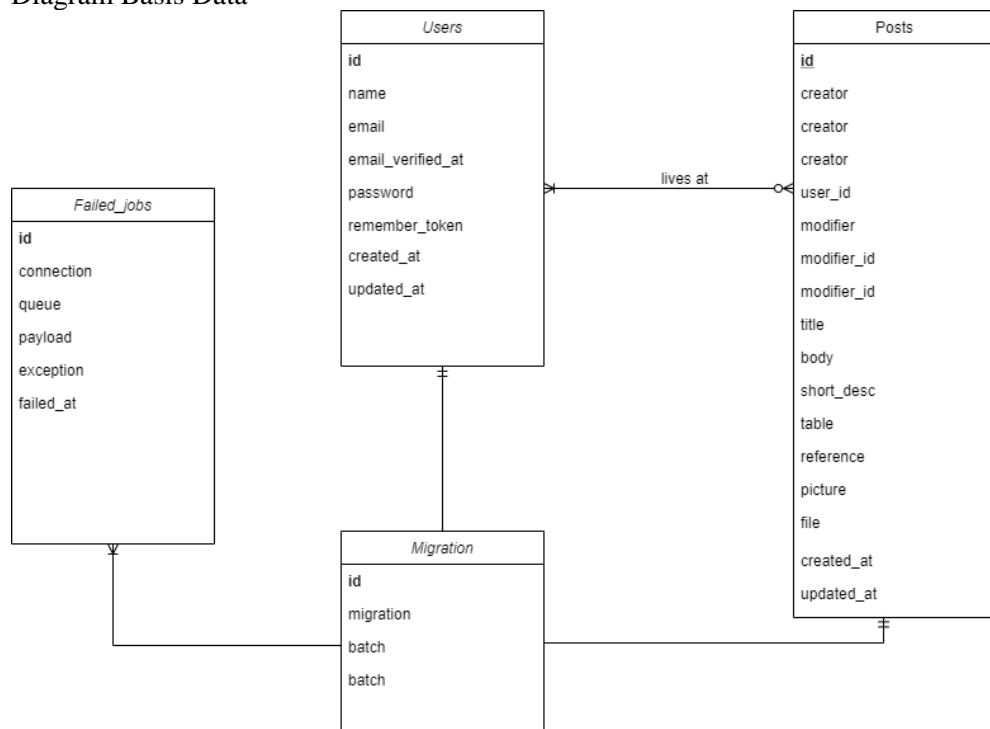


Gambar 2. Struktur Dalam Situs

Diagram diatas menjelaskan bagaimana struktur dalam situs berbasis *Laravel* berjalan. Deskripsi dari diagram diatas adalah:

- Jika pengguna mengakses alamat situs, maka situs akan mengarahkan pengguna (*route*) ke halaman yang diatur dalam *controller*. *Controller* ini akan memberikan halaman situs beserta dengan data yang diperlukan dalam bentuk *view*.
- View* ini dituliskan dalam bahasa *PHP* yang dibantu dengan format *Blade*.
- Data yang disampaikan oleh *controller* diambil dari *database* dan diatur bahasa pengambilan datanya menggunakan aplikasi *Eloquent Object Relational Mapping* yang disediakan oleh *Laravel*. Aplikasi ini sangat membantu karena penggunaanya tidak perlu lagi menuliskan pengambilan data dalam bentuk *Structured Query Language (SQL)*.

3. Diagram Basis Data



Gambar 3. Rancangan Basis Data

Database yang dibuat ada 2, yaitu *Posts* dan *Users*. Database tersebut dibuat secara otomatis menggunakan *migration* pada *Laravel* dan secara otomatis membuat database *Migration* dan *Failed_Jobs* apabila proses migrasi terjadi kesalahan. Fungsi migrasi ini dipakai untuk memberikan kemudahan *roll-back* apabila terjadi masalah saat pembuatan database.

4. Proses Ekstraksi Dokumen

Aplikasi ini juga disediakan fungsi untuk ekstraksi dokumen yang terintegrasi ke dalam fungsi pengunggahan dokumen. Fungsi ini mengimplementasi sebuah aplikasi *open-source* “*PDF Parser*”. Cara kerja aplikasi ini adalah sebagai berikut:

- Di dalam *controller*, dibuat validasi apakah pada saat pembuatan disediakan dokumen *pdf* atau tidak. Jika iya maka akan dokumen akan diproses.
- Selanjutnya dokumen tersebut akan divalidasi apakah dokumen tersebut merupakan *pdf* atau tidak. Jika iya akan diproses lebih lanjut, jika tidak maka pengguna akan diberikan pemberitahuan *error*.
- Selanjutnya dokumen diproses menggunakan fungsi yang diberikan oleh *PDF Parser*:

```

// Parse pdf file and build necessary objects.
$parser = new \Smalot\PdfParser\Parser();
$pdf = $parser->parseFile('document.pdf');

```

Gambar 4. Penggunaan Aplikasi *PDF Parser*

- Kemudian teks yang dihasilkan akan diproses untuk menghilangkan tabulasi dan spasi yang berlebih. Kemudian dokumen akan disimpan dengan nama unik dalam *server* dan *database*.

-
- e. Apabila dalam proses terjadi kesalahan, maka akan diberikan peringatan kepada pengguna bahwa proses upload gagal.

3.2 Implementasi Antar Muka

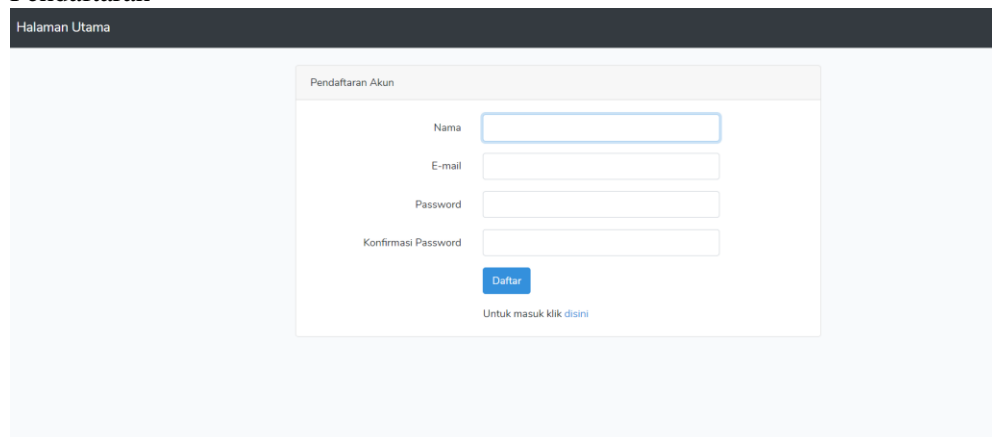
Berikut ini adalah hasil implementasi antar muka dari situs yang dikembangkan setelah proses desain dilakukan:

1. Landing Page



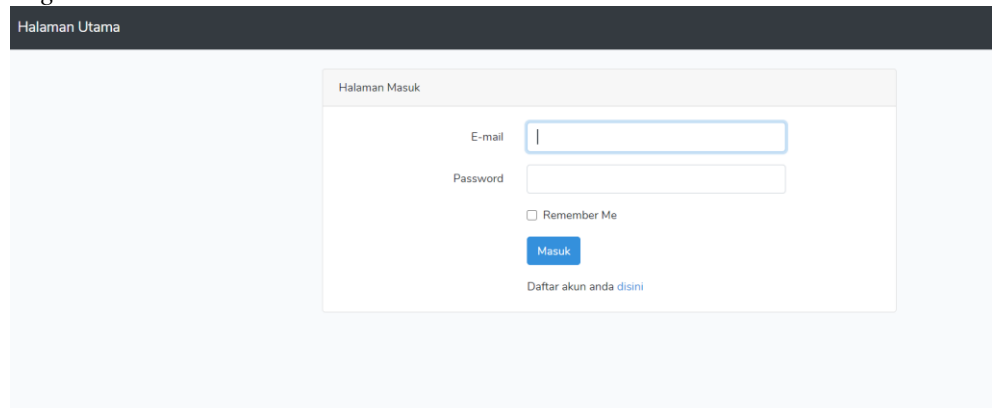
Gambar 5. Tampilan Halaman *Landing Page*

2. Pendaftaran



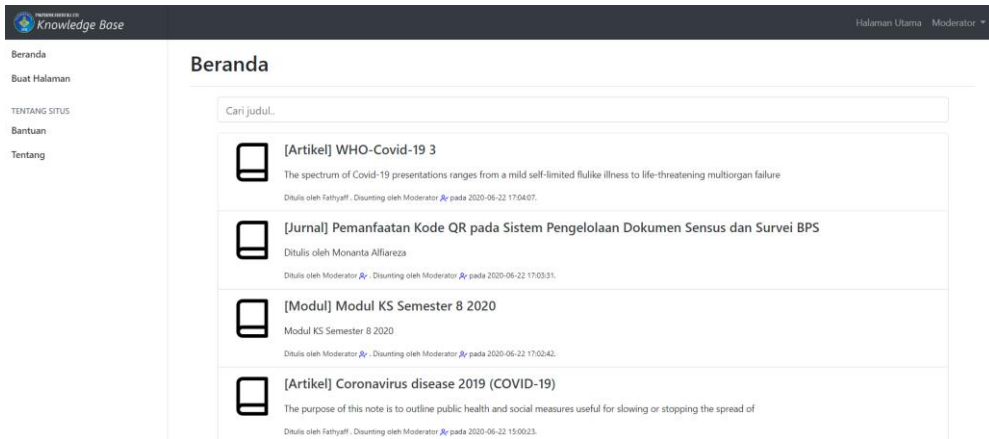
Gambar 6. Tampilan Halaman Pendaftaran

3. Login



Gambar 7. Tampilan Halaman *Login*

4. Beranda



Gambar 8. Tampilan Halaman Beranda

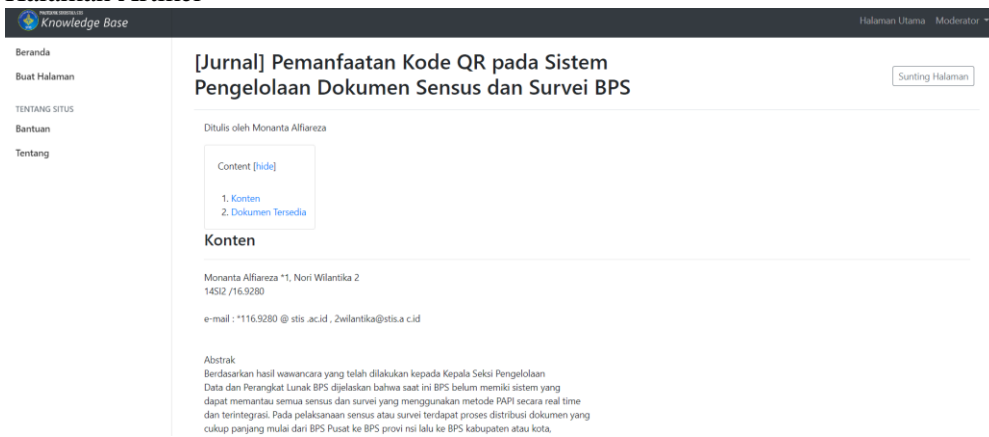
Keterangan: Pengguna yang tidak *login* akan melihat tombol *login* pada pojok kanan atas halaman.

Di halaman beranda, pengguna juga dapat melakukan pencarian judul artikel yang sudah ada. Pengguna juga diberikan kenyamanan dalam pencarian dengan diberikan fitur *suggestion* pada kolom pencarian.



Gambar 9. Tampilan Hasil Pencarian

5. Halaman Artikel



Gambar 10. Tampilan Halaman Artikel

Keterangan: Pengguna yang tidak *login* tidak bisa menyunting halaman. Pengguna yang membuat artikel juga satu-satunya yang bisa menghapus halaman tersebut.

6. Pembuatan Halaman

The screenshot shows the 'Buat Halaman' (Create Page) interface. The top navigation bar includes the 'Knowledge Base' logo and links for 'Halaman Utama' and 'Moderator'. The left sidebar contains links for 'Beranda', 'Buat Halaman', 'TENTANG SITUS', 'Bantuan', and 'Tentang'. The main content area is titled 'Buat Halaman' and contains a form with the following fields: 'Judul' (Title), 'Keterangan Singkat' (Short Description), and a large text area for 'Penjelasan singkat tentang artikel' (Short explanation of the article). The text area is powered by TinyMCE. Below the text area, there is a 'Submit' button and a 'Choose File' button. A note indicates that the text must be within 131 characters and that the file must be a PDF document.

Gambar 11. Tampilan Pembuatan Halaman

7. Penyuntingan Halaman

The screenshot shows the 'Sunting Halaman' (Edit Page) interface. The top navigation bar includes the 'Knowledge Base' logo and links for 'Halaman Utama' and 'Moderator'. The left sidebar contains links for 'Beranda', 'Buat Halaman', 'TENTANG SITUS', 'Bantuan', and 'Tentang'. The main content area is titled 'Sunting Halaman' and contains a form with the following fields: 'Judul' (Title), 'Keterangan Singkat' (Short Description), and a large text area for 'Ditulis oleh Monanta Alfiazeza' (Written by Monanta Alfiazeza). The text area is powered by TinyMCE. Below the text area, there is a 'Choose File' button and a 'File currently attached: 16.9280-makalah_1592456034.pdf' label. A 'Cancel Edit Page' button is located in the top right corner.

Gambar 12. Tampilan Penyuntingan Halaman

8. Bantuan

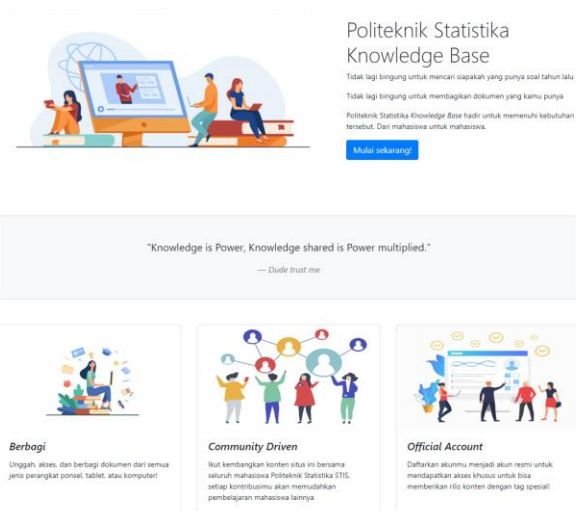
The screenshot shows the 'Bantuan' (Help) interface. The top navigation bar includes the 'Knowledge Base' logo and links for 'Halaman Utama' and 'Moderator'. The left sidebar contains links for 'Beranda', 'Buat Halaman', 'TENTANG SITUS', 'Bantuan', and 'Tentang'. The main content area is titled 'Bantuan' and contains a list of links: 'Membuat Halaman Baru', 'Menyunting Halaman', and 'Pengaturan Profil'. The 'Membuat Halaman Baru' link is highlighted. To the right of the links, there is a list of instructions for creating a new page.

- Pilih "Buat Halaman"
- Isikan Judul dan deskripsi singkat berisikan keterangan dari artikel dibuat
- Pilih file PDF yang ingin diunggah, file tersebut akan otomatis dibaca dan dijadikan isi utama artikel tersebut
- Sunting seperlunya agar teks mudah dibaca
- Klik tombol "Submit"

Gambar 13. Tampilan Halaman Bantuan

9. Tentang Situs

Beranda
Buat Halaman
TENTANG SITUS
Bantuan
Tentang



Gambar 14. Tampilan Halaman Tentang Situs

3.3 Hasil Pengujian

1. White Box

Uji *white box* dilakukan menggunakan fitur yang disediakan oleh *Laravel* yaitu fitur *testing*. Pengujian dilakukan dengan tujuan untuk mengecek apakah setiap *routing* yang tersedia dapat diakses dan halaman yang diperlukan *login* dapat diakses hanya oleh pengguna yang memegang *logged-in*. Hasil uji menghasilkan *assertion results* yang berguna bagi penguji untuk memperbaiki *routing* tersebut.

2. Black Box

Uji *black box* ditujukan kepada 15 calon pengguna yang merupakan mahasiswa Politeknik Statistika STIS. Hasil pengujian menghasilkan banyak *bugs* yang sangat berguna untuk menyempurnakan situs agar pengalaman pengguna dalam situs semakin baik. Setelah proses pembaharuan, uji dilakukan lagi kepada 5 orang yang mengalami *error* dan *bugs* yang menghasilkan hasil yang diinginkan dari scenario pengujian yang diberikan.

3. Survei Usability Scale (SUS)

SUS dilakukan kepada 13 pengguna yang merupakan mahasiswa Politeknik Statistika STIS. Survei ini dilakukan menggunakan aplikasi yang disediakan oleh *usabilitytest.com*. Data dari hasil yang diisi oleh pengguna adalah:

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	SUS	Usability	Learnability
3	4	5	4	5	2	2	2	1	4	50.0	56.3	25.0
4	1	4	1	4	2	4	2	5	2	82.5	81.3	87.5
3	1	4	1	3	3	5	2	4	1	77.5	71.9	100.0
4	3	4	2	3	4	5	3	4	2	65.0	62.5	75.0
3	3	5	2	4	2	5	3	4	2	72.5	71.9	75.0
3	2	3	2	4	3	4	4	3	1	62.5	56.3	87.5
4	3	4	2	5	1	3	1	5	1	82.5	81.3	87.5
4	2	4	1	4	2	4	2	4	1	80.0	75.0	100.0
3	2	4	1	4	2	4	2	5	2	77.5	75.0	87.5
5	2	5	1	4	2	5	1	4	2	87.5	87.5	87.5
4	4	3	2	3	2	4	3	5	1	67.5	62.5	87.5
4	3	5	2	4	2	4	2	4	3	72.5	75.0	62.5

GRADE ① 68% percentile	PARTICIPANTS ① complete responses: 12	SUS score:	73.1	①
		Learnability:	80.2	①
		Usability:	71.4	①
		Standard deviation:	10.5	①
		Cronbach's alpha:	0.644	①

Gambar 15. Tabel Hasil SUS

SUS score merepresentasikan nilai dari total pengguna dapat menggunakan dan mempelajari kegunaan situs. Nilai yang di dapat adalah 73.1 yang berarti lebih baik daripada rata-rata nilai *SUS* yaitu 68. Hasil ini mendapatkan *grade* “C” atau “good” karena nilai berada dalam interval 70-79 dan sudah lebih baik daripada 67% hasil *SUS* lainnya (Bangor et al, 2009).

Learnability merupakan nilai yang merepresentasikan apakah pengguna membutuhkan bantuan orang lain untuk dapat mengoperasikan situs. Nilai yang di dapat oleh *SUS* ini adalah 80.2, yang berarti “excellent”.

Usability merupakan nilai yang merepresentasikan seberapa mudah bagi pengguna untuk menjalankan fungsi-fungsi yang disediakan situs dan seberapa efektif pengguna menjalankannya. Skor yang di dapat adalah 71.4, yang berarti “good” atau sudah baik dalam tingkat *usability*.

Standard deviation merupakan nilai yang merepresentasikan seberapa jauh nilai yang dihasilkan dengan rata-rata nilai *SUS* secara umum. Nilai yang dihasilkan adalah 10.5, apabila dihitung *Coefficient of Variation* akan menghasilkan nilai 0.144 yang berarti data memiliki standar deviasi yang rendah dan sampel memiliki keserupaan satu sampel dengan sampel lainnya.

Cronbach's alpha merupakan nilai yang merepresentasikan seberapa terkait sampel yang diambil sebagai satu jenis grup. Nilai yang dihasilkan adalah 0.644, lebih rendah dari nilai minimum 0.7. Hal ini berarti sampel yang diambil memiliki perbedaan sifat dalam menggunakan aplikasi.

Jika nilai *SUS* di jabarkan per pertanyaan, maka dapat dilihat untuk pertanyaan nomor 4 dan 10 memiliki skor rata-rata dari seluruh sampel:

1. Q4 : “I think that I would need the support of a technical person to be able to use this application.”
2. Q10 : “I needed to learn a lot of things before I could get going with this application.”

Kedua pertanyaan tersebut memiliki skor rata-rata sebesar 1.75 dan 1.833 dari total 5 untuk sempurna. Hal ini berarti rata-rata pengguna memerlukan bantuan teknis dan perlu belajar lebih untuk membiasakan diri ketika memakai aplikasi.

4. KESIMPULAN

Berdasarkan penelitian yang sudah dilakukan dengan mempertimbangkan hasil timbal balik yang dihasilkan dari pengujian, dapat disimpulkan:

1. Implementasi sistem mampu diterima dengan baik oleh target pengguna. Hal ini disimpulkan melalui hasil *SUS* dan umpan balik yang diterima dari pengguna saat melakukan *black box testing*.
2. Fitur-fitur utama (penyimpanan, pencarian, dan pembagian) sudah dijalankan dengan baik. Terutama fitur *auto-complete* atau *suggestion* dalam pencarian yang paling menantang untuk diimplementasi, namun memiliki efek yang baik demi kenyamanan pengguna. Hal ini disimpulkan dari sebagian besar umpan balik pada saat *black box testing* dimana pengguna menyarankan untuk diadakan fungsi pencarian untuk mempermudah akses pengguna ke artikel yang pengguna cari. Fungsi ini terinspirasi dari penelitian yang dilakukan oleh Sanjaya Abdillah (2018).
3. Pengguna membutuhkan bantuan dari teknisi dan perlu untuk membiasakan diri untuk menggunakan aplikasi. Hal ini disimpulkan melalui skor terendah dari setiap sampel *SUS* yaitu pada pertanyaan nomor 4 dan 10, masing-masing memiliki rata-rata skor 1.75 dan 1.833. Hal ini juga berarti bahwa implementasi desain antar muka tidak sepenuhnya intuitif untuk pengguna dan memerlukan pengguna untuk menggunakan aplikasi secara berkala untuk terbiasa dengan antar muka situs.

Ada pula beberapa hal yang kurang dalam penelitian ini dan dapat dipertimbangkan untuk penelitian kedepan mengenai pengembangan aplikasi *knowledge management* seperti:

1. Implementasi fitur *draft*, ketika pengguna mencoba menyunting artikel orang lain. Maka hasil suntingan tersebut akan ditunda untuk diterima oleh moderator atau penulis artikel terima dan diimplementasi. Karena saat ini setiap pengguna mampu menyunting milik orang lain secara langsung.
2. Dikembangkannya fungsi ekstraksi teks dan gambar dari sebuah pdf, karena pada penelitian ini aplikasi hanya mampu mengambil teks.
3. Implementasi fitur *tagging*, untuk mempermudah pengguna dalam mengkategorisasi dan mengklasifikasikan artikel.
4. Implementasi fitur diskusi agar pengguna mampu berdiskusi untuk memberi tahu pengguna lain hal yang berkaitan dengan situs secara umum atau artikel.

DAFTAR PUSTAKA

- Abdillah, K. Sanjaya. (2018). *Perancangan Prototipe Knowledge Management System*. Sekolah Tinggi Ilmu Statistik.
- Bangor, A., Kortum, P., & Miller, J. (2009). *Determining what individual SUS scores mean: Adding an adjective rating scale*. Journal of usability studies, 4(3), 114-123.
- Bassil, Y. (2012). *A simulation model for the waterfall software development life cycle*. arXiv preprint arXiv:1205.6904.
- GNU. *GNU GENERAL PUBLIC LICENSE*. <https://www.gnu.org/licenses/gpl-3.0.en.html>. (diakses pada 15 Juni 2020).
- Lewis, J. R., & Sauro, J. (2009, July). *The factor structure of the system usability scale*. In *International conference on human centered design* (pp. 94-103). Springer, Berlin, Heidelberg.
- Mohajan, H. (2016). *A Comprehensive Analysis of Knowledge Management Cycles*.
- Murnane, T., & Reed, K. (2001, August). *On the effectiveness of mutation analysis as a black box testing technique*. In *Proceedings 2001 Australian Software Engineering Conference* (pp. 12-20). IEEE.
- Nidhra, S., & Dondeti, J. (2012). *Black box and white box testing techniques-a literature review*. International Journal of Embedded Systems and Applications (IJESA), 2(2), 29-50.
- O'Dell, C., & Grayson, C. J. (1998). *If only we knew what we know: the transfer of internal knowledge and best practice*. New York: Free Press.
- Ruparelia, N. B. (2010). *Software development lifecycle models*. ACM SIGSOFT Software Engineering Notes, 35(3), 8-13.
- Shao, D., Khurshid, S., & Perry, D. E. (2007). *A case for white-box testing using declarative specifications poster abstract*. In *Testing: Academic and Industrial Conference Practice and Research Techniques-MUTATION (TAICPART-MUTATION 2007)* (pp. 137-137). IEEE.
- University of North Carolina. *Introduction to Knowledge Management*. www.unc.edu. Chapel Hill. Archived from the original on March 19, 2007. (diakses pada 15 Juni 2020).