BAB II. LANDASAN TEORI

Pada bab ini akan diuraikan kajian pustaka dan dasar teori yang mendukung penelitian ini. Dasar teori tersebut diperoleh dari beberapa referensi yang relevan dengan topik yang diangkat dalam penelitian ini. Dalam bab ijni akan dijelaskan mengenai kajian pustaka dan metode *Longest Common Subsequence* (LCS) yang merupakan topik utama diadakannya penelitian ini.

2.1 Penelitian Terdahulu

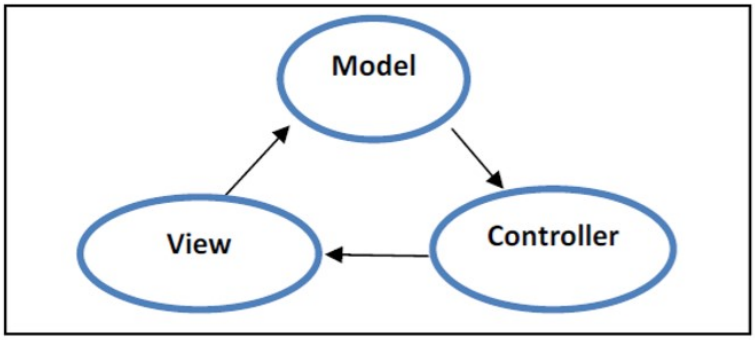
Berdasarkan hasil penelitian terdahulu, ditemukan penelitian sejenis tentang deteksi kemiripan dalam referensi [1] yang membahas mengenai deteksi plagiarisme pada judul skripsi, metode yang digunakan adalah algoritma Winnowing. Pada penelitian tersebut, aplikasi yang dibangun dapat memberikan informasi berupa hasil plagiarisme pada judul skripsi. Namun, pengujian yang dapat dilakukan pada penelitian tersebut, terbatas pada teks judul skripsi.

Penelitian yang dilakukan dalam referensi [2] mengenai deteksi kemiripan antara dua dokumen dengan dilakukan perbandingan pattern pada dokumen teks, metode yang digunakan untuk mendeteksi kemiripan pada dokumen adalah Algoritma Rabin Karp. Penelitian ini berhasil menciptakan aplikasi deteksi plagiarisme yang bekerja dengan baik serta memperlihatkan hasil similarity-nya. Namun, pada penelitian ini hanya dapat melakukan pengujian antara dua dokumen.

Berdasarkan kedua referensi yang sudah dijelaskan tersebut, maka pada penelitian skripsi ini permasalahan yang dibahas adalah pembuatan aplikasi pendeteksi plagiarisme pada dokumen tugas akhir atau skripsi berdasarkan tingkat kesamaan dengan menggunakan metode *Longest Common Subsequence* untuk mencari nilai *similarity* yang terdapat pada satu atau lebih dari satu kandidat pembanding yang dibandingkan. Persamaan penelitian ini dengan kedua referensi penelitian tersebut yaitu sistem pendeteksi kemiripan pada dokumen.

2.2 Model View Controller (MVC)

Menurut [17], *Model View Controller* (MVC) merupakan *framework* yang menerapkan model MVC, model ini diterapkan guna dapat memberi dampak kedisiplinan dalam menuliskan kode program serta mempermudah dalam pengembangan aplikasi karena telah terstruktur dengan baik, model MVC dibagi menjadi 3 bagian yaitu : model, *view,* dan *controller* seperti pada gambar 2.1.



Gambar 2. 1 Model View Controller

Bagian-bagian tersebut memiliki arti sebagai berikut:

- *Model* merupakan bagian yang berinteraksi dengan *database*.
- *View* merupakan *pages* yang tampil di halaman *web*.
- *Controller* akan mengatur interaksi antar *pages*/ mengkontrol hubungan *view* dan *model*.

Keuntungan MVC adalah sebagai berikut:

- *Division/testability*

  Cotrollers mengatur interaksi antara *User Interface* dan data (*model*).
- *Flexibility*

  Setiap individual *layer* mudah dibuang tanpa mempengaruhi *layer* lainnya karena dapat di *costumize* sehingga *user interface* dapat diganti atau menggunakan *database* yang berbeda.
- *Maintability*

Walaupun dapat di *costumize*, tapi *projects* harus tetap memiliki code yang teratur.

2.3 Metode *Longest Common Subsequence*

Longest Common Subsequences merupakan masalah dalam mencari *subsequence* terpanjang dari beberapa *sequence* (biasanya hanya 2 buah *sequence*), dimana sebuah *subsequence* merupakan sekumpulan kata dari *sequence* yang urutan kemunculannya sama [7]. *Subsequence* dari sebuah *string* A adalah sekumpulan kata yang ada pada A dengan urutan kemunculan kata yang sama.

Dalam rangkaian Z yang merupakan *subsequence* dari X (sebagai *sequence*) dimana X = $<x_1,x_2,\ldots,x_m>$, jika terdapat urutan menaik $<i_1,i_2,\ldots,i_n>$ yang merupakan indeks X untuk semua j=1,2,…,k, yang memenuhi $x_i=z_j$. misal pada *common subsequences* dari 2 rangkaian string1 = tadi saya sedang makan dikantin sekolah pada jam istirahat dan string2 = saya sedang mengerjakan tugas seni budaya pada jam istirahat, maka saya sedang, dan pada jam istirahat adalah *common subsequences* dari string1 dan string2. *Longest Common Subsequences* adalah *common subsequences* dari rangkaian hasil yang paling panjang pada 2 rangkaian *string* (string1 dan string2), sehingga *Longest Common Subsequences* yang terpilih adalah pada jam istirahat.

2.4.1 Mendapatkan Panjang LCS

Untuk menjabarkan dalam mempermudah dalam menjelaskan, akan dijabarkan sebagai berikut :

Misal *m* adalah panjang string1, *n* adalah panjang string2, dan tab adalah matriks berukuran *m* x *n,* dan tab[*i, j*] adalah panjang LCS dari *subsequences* pertama yang terdiri dari *i* kata pada string1 dengan *subsequences* kedua yang terdiri dari *j* kata pada string2, maka untuk $1 \le i \le m$ dan $1 \le j \le n,$ sesuai dengan fungsi rekursif diatas, maka :

$$\text{tab}[i, j] = \begin{cases} \text{tab}[i-1, j-1] + 1 , S1[i] = s2[j] \\ \max(\text{tab}[i-1, j], \text{tab}[i, j-1]) \end{cases}$$

Teks yang dibentuk yaitu string1 = tadi saya sedang makan dikantin sekolah pada jam istirahat dan string2 = saya sedang mengerjakan tugas seni budaya pada jam istirahat, tabel matrik yang terbentuk berukuran 9 x 7 yang isinya sebagai berikut :

Tabel 2. 1 Hasil Kalkulasi LCS

		1	2	3	4	5	6	7	8	9
		tadi	saya	sedang	makan	dikantin	sekolah	pada	jam	istirahat
1	saya	0	1	1	1	1	1	1	1	1
2	sedang	0	1	2	2	2	2	2	2	2
3	mengerjakan	0	1	2	2	2	2	2	2	2
4	tugas	0	1	2	2	2	2	2	2	2
5	pada	0	1	2	2	2	2	3	3	3
6	jam	0	1	2	2	2	2	3	4	4
7	istirahat	0	1	2	2	2	2	3	4	5

Dan sesuai dengan arti notasi dari setiap tab[*i, j*] yang berisi pada tabel 2.1 bahwa LCS dari string1 dan string2 adalah tab[*m, n*] atau dalam kasus ini adalah ditemukan 5 kata.

2.4.2 Mendapatkan LCS

Sedangkan untuk mendapatkan LCS yang dimaksud, kita bisa merunut balik pada tabel 2.1 yang dimulai dari tab[*m, n*]. Dalam runut balik ini, yang kita lakukan sebenarnya hanyalah merunut balik pilihan yang dilakukan pada saat kalkulasi tab[i, j].

Untuk setiap $1 \leq i \leq m$ dan $1 \leq j \leq n$, jika string1[i] sama dengan string2[j], maka kata itu pasti terdapat pada LCS. Selain itu, cek, dari mana mendapatkan LCS saat itu, lakukan pemilihan yang sama. Lakukan hal tersebut dimulai dari $i = m$ dan $j = n$. Setelah hal itu dilakukan maka akan terbentuk tabel di bawah ini. Di mana cell yang dihitamkan menandakan kata yang masuk ke dalam LCS.

Tabel 2. 2 Hasil Penurutan Balik Untuk Mendapatkan LCS

		1	2	3	4	5	6	7	8	9
		tadi	saya	sedang	makan	dikantin	sekolah	pada	jam	istirahat
1	Saya	0	1	1	1	1	1	1	1	1
2	Sedang	0	1	2	2	2	2	2	2	2
3	Mengerjakan	0	1	2	2	2	2	2	2	2
4	Tugas	0	1	2	2	2	2	2	2	2
5	Pada	0	1	2	2	2	2	3	3	3
6	Jam	0	1	2	2	2	2	3	4	4
7	Istirahat	0	1	2	2	2	2	3	4	5

Berdasarkan pada tabel 2.2 dapat dilihat bahwa hasil LCS yang ditemukan adalah saya sedang, pada jam istirahat.

2.4 Codeigniter

CodeIgniter adalah *framework* open source yang powerful yang dibangun menggunakan bahasa pemrograman PHP yang terkenal sebagai PHP *framework* yang mudah dikuasai [15], codeigneter dibangun untuk PHP programmers yang membutuhkan toolkit sederhana dan untuk membuat full-featured web applications. CodeIgniter menggunakan konsep MVC *framework* yang di *design* untuk mempermudah pengguna dan ketika dalam pengembangan aplikasi. Pada penelitian ini codeigniter digunakan untuk membangun aplikasi yang berjudul deteksi plagiarisme pada dokumen skripsi berdasarkan tingkat kesamaan dengan menggunakan metode *Longest Common Subsequence.*

2.5 Bootstrap

Bootsrap adalah *framework* css yang dapat digunakan untuk membangun tampilan *web* [16]. *Framework* ini menggunakan coding CSS dan JavaScript, namun tetap bisa membuat *website* yang powerfull mengikuti perkembangan *browser*. *Website* yang menggunakan bootstrap akan menjadi *website* yang fleksibel dengan penataan *layout* standard dari bootstrap yang dinamakan grid, sehingga nyaman dan tentu saja cepat baik dalam proses pembuatan maupun ketika *maintenance.* Tidak hanya penataan layoutnya saja yang mudah, bootstrap juga menyediakan komponen-komponen yang siap digunakan untuk mendesain sebuah template.

Komponen yang disediakan sangat banyak dan memiliki beberapa variasi, seperti *dropdowns, button, input, navbar, label, panels, alerts* dan masih banyak lagi yang dapat di *explore* melalui *website* resminya https://getbootstrap.com. Pada penelitian ini desain templatenya menggunakan bootstrap versi 3.

2.6 MySQL

MySQL adalah *database product* yang bersifat *open source* [19]. MySQL digunakan secara gratis dan hanya menghabiskan sedikit biaya bagi periklanannya namun sangat populer. Tidak seperti *commercial database*, MySQL sangat terjangkau dan mudah digunakan.

Fitur yang terdapat di MySQL antara lain:

- *Openess*

  MySQL sangat terbuka dalam setiap hubungan. SQL *dialect* yang digunakan adalah ANSI SQL2 sebagai pembangunnya. *Database engine* menjalankan *platform* yang tidak terhitung, termasuk Windows 2000, Mac OS X, Linux, FreeBSD, dan Solaris. Jika *binary* tidak tersedia untuk *platform*, user dapat mengaksesnya dari *source* untuk di *compile* ke *platform* tersebut.

- *Application Support*

  MySQL memiliki API untuk semua bahasa pemrograman. *User* bisa menulis aplikasi *database* yang mengakses MySQL di C, C++, Eiffel, Java, Perl, PHP, Phyton, dan Tcl.

- *Cross-database joins*

  User bisa mengkonstruksi MySQL queries yang bisa menggabungkan tabel dari database yang berbeda.

- *Outer join support*

  MySQL mendukung kiri dan kanan outer joins menggunakan ANSI dan sintaks ODBC.

2.7 Kalimat

Menurut [12, pp. 317-319] Kalimat adalah satuan bahasa terkecil, dalam wujud lisan atau tulisan, yang mengungkapkan pikiran yang utuh. Dalam wujud lisan, kalimat diucapkan dengan suara naik turun dan keras lembut, disela jeda dan diakhiri dengan intonasi akhir yang diikuti oleh kesenyapan yang mencegah terjadinya perpaduan ataupun asimilasi bunyi ataupun proses fonologis lainnya.

Dilihat dari segi bentuknya, kalimat dapat dirumuskan sebagai konstruksi sintaksis terbesar yang terdiri atas dua kata atau lebih. Hubungan structural antara kata dan kata, atau kelompok kata dan kelompok kata yang lain, berbeda-beda. Sementara itu, kedudukan tiap kata atau kelompok kata dalam kalimat itu berbeda-beda pula. Ada kata atau kelompok kata yang dapat dihilangkan dengan menghasilkan bentuk yang tetap berupa kalimat (1b), dan ada pula yang tidak seperti pada (2b).

1. a. Ibu pergi ke pasar.
   b. Ibu pergi.
2. a. Masalah itu menyangkut masa depan kita.
   b. Masalah itu menyangkut.

2.8 Diagram *Use Case*

Use case diagram adalah model diagram dari *UML (Unified Modeling Language)* yang digunakan untuk menggambarkan permintaan secara fungsional yang diharapkan dari sebuah sistem. *Use case* diagram menekankan pada "siapa" melakukan "apa" dalam lingkungan sistem perangkat lunak akan dibangun. *Use-case* diagram sebenarnya terdiri dari dua bagian besar; yang pertama adalah *use case* diagram (termasuk gambar use case *dependencies*) dan *use case description*.

Use case diagram merupakan gambaran graphical dari beberapa atau semua *actor*, *use case*, dan interaksi [13] diantara komponen-komponen tersebut yang memperkenalkan suatu sistem yang akan dibangun. *Use case* diagram menjelaskan manfaat suatu sistem jika dilihat menurut pandangan orang yang berada di luar sistem. Diagram ini menunjukkan fungsionalitas suatu sistem atau kelas dan bagaimana sistem tersebut berinteraksi dengan dunia luar.

2.9 *Framework*

Menurut [17], dengan menggunakan framework, kita tidak perlu membuat program dari awal, tetapi kita sudah diberikan library fungsi-fungsi yang sudah diorganisasikan untuk dapat membuat suatu program dengan cepat. Dalam [18], framework merupakan kerangka kerja yang memudahkan programmer untuk membuat sebuah aplikasi sehingga programmer akan lebih mudah melakukan perubahan (customize) terhadap aplikasinya dan dapat memakainya kembali untuk aplikasi lain yang sejenis. Berdasakan penjelasan di atas framework merupakan kerangka kerja yang memudahkan programmer untuk membuat aplikasi dengan library fungsi-fungsi yang sudah diorganisasikan untuk dapat membuat suatu program dengan cepat.