

**SKRIPSI**

**SISTEM PENILAIAN SIDANG SKRIPSI 2 DENGAN  
ANGULARJS**



**BILLY YANUAR**

**NPM: 2012730017**

**PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS  
UNIVERSITAS KATOLIK PARAHYANGAN**

**«tahun»**



**UNDERGRADUATE THESIS**

**«JUDUL BAHASA INGGRIS»**



**BILLY YANUAR**

**NPM: 2012730017**

**DEPARTMENT OF INFORMATICS  
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES  
PARAHYANGAN CATHOLIC UNIVERSITY  
«tahun»**



**LEMBAR PENGESAHAN**

**SISTEM PENILAIAN SIDANG SKRIPSI 2 DENGAN  
ANGULARJS**

**BILLY YANUAR**

**NPM: 2012730017**

**Bandung, «tanggal» «bulan» «tahun»**

**Menyetujui,**

**Pembimbing Utama**

**Pembimbing Pendamping**

**«pembimbing utama/1»  
Ketua Tim Penguji**

**«pembimbing pendamping/2»  
Anggota Tim Penguji**

**«penguji 1»**

**«penguji 2»**

**Mengetahui,**

**Ketua Program Studi**

**Thomas Anung Basuki, Ph.D.**



## PERNYATAAN

Dengan ini saya yang bertandatangan di bawah ini menyatakan bahwa skripsi dengan judul:

### **SISTEM PENILAIAN SIDANG SKRIPSI 2 DENGAN ANGULARJS**

adalah benar-benar karya saya sendiri, dan saya tidak melakukan penjiplakan atau pengutipan dengan cara-cara yang tidak sesuai dengan etika keilmuan yang berlaku dalam masyarakat keilmuan.

Atas pernyataan ini, saya siap menanggung segala risiko dan sanksi yang dijatuhkan kepada saya, apabila di kemudian hari ditemukan adanya pelanggaran terhadap etika keilmuan dalam karya saya, atau jika ada tuntutan formal atau non-formal dari pihak lain berkaitan dengan keaslian karya saya ini.

Dinyatakan di Bandung,  
Tanggal «tanggal» «bulan» «tahun»

Meterai

Billy Yanuar  
NPM: 2012730017





## ABSTRAK

«Tuliskan abstrak anda di sini, dalam bahasa Indonesia» Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetur.

**Kata-kata kunci:** «Tuliskan di sini kata-kata kunci yang anda gunakan, dalam bahasa Indonesia»



## ABSTRACT

«Tuliskan abstrak anda di sini, dalam bahasa Inggris» Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

**Keywords:** «Tuliskan di sini kata-kata kunci yang anda gunakan, dalam bahasa Inggris»



*«kepada siapa anda mempersembahkan skripsi ini...?»*



## KATA PENGANTAR

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Bandung, «bulan» «tahun»

Penulis





## DAFTAR ISI

<b>KATA PENGANTAR</b>	<b>xv</b>
<b>DAFTAR ISI</b>	<b>xvii</b>
<b>DAFTAR GAMBAR</b>	<b>xviii</b>
<b>DAFTAR TABEL</b>	<b>xix</b>
<b>1 PENDAHULUAN</b>	<b>1</b>
1.1 Latar Belakang . . . . .	1
1.2 Rumusan Masalah . . . . .	2
1.3 Tujuan . . . . .	2
1.4 Batasan Masalah . . . . .	2
1.5 Metode Penelitian . . . . .	2
1.6 Sistematika Penulisan . . . . .	3
<b>2 DASAR TEORI</b>	<b>5</b>
2.1 CodeIgniter . . . . .	5
2.1.1 Flowchart Aplikasi CodeIgniter . . . . .	5
2.1.2 Model-View-Controller . . . . .	6
2.2 AngularJS . . . . .	6
2.2.1 Gambaran Konseptual . . . . .	7
2.2.2 Data Binding . . . . .	7
2.3 Twitter Bootstrap . . . . .	8
<b>3 ANALISIS</b>	<b>9</b>
3.1 Analisis Data Penilaian Skripsi . . . . .	9
3.2 Analisis Tampilan Sistem Informasi Penilaian Skripsi . . . . .	9
<b>4 PENUTUP</b>	<b>11</b>
4.1 Kesimpulan . . . . .	11
<b>DAFTAR REFERENSI</b>	<b>13</b>
<b>A FORM PENILAIAN SKRIPSI</b>	<b>15</b>
<b>B THE SOURCE CODE</b>	<b>17</b>

## DAFTAR GAMBAR

2.1	Flowchart CodeIgniter . . . . .	5
2.2	Data Binding Classical Templates System . . . . .	8
2.3	Data Binding pada Angular . . . . .	8
3.1	Perbandingan Tampilan [?] . . . . .	10
3.2	Perkiraan Tampilan [?] . . . . .	10
A.1	Form Penilaian Skripsi saat sidang . . . . .	15
A.2	Form Rekapitulasi Penilaian Skripsi saat sidang . . . . .	16

## DAFTAR TABEL



# BAB 1

## PENDAHULUAN

### 1.1 Latar Belakang

Program Studi Teknik Informatika di Universitas Katolik Parahyangan memiliki beberapa syarat kelulusan antara lain minimal sks yang lulus 144 yang terdiri dari matakuliah wajib dan pilihan, indeks prestasi minimum adalah 2.00 dengan maksimum 14 semester. Salah satu matakuliah wajib yang harus ditempuh dan lulus adalah skripsi. Skripsi di Program Studi Teknik Informatika di Universitas Katolik Parahyangan dibagi menjadi 2 matakuliah yaitu skripsi 1 dan skripsi 2.

Sistem penilaian sidang skripsi 2 pada Program Studi Teknik Informatika di Universitas Katolik Parahyangan masih bersifat manual dimana penilai mengisi data-data mahasiswa memberikan nilai untuk mahasiswa pada saat sidang dan juga melakukan penghitungan bobot nilai total.

Sifat manual ini mengakibatkan kelalaian manusia dalam melakukan penilaian pun beberapa kali tidak dapat dihindarkan. Kelalaian manusia yang biasa terjadi contohnya adalah kesalahan perhitungan nilai akhir oleh penilai, kesalahan penulisan nama dan NPM mahasiswa yang bersangkutan, kesalahan penulisan semester atau tahun ajaran saat penilaian skripsi<sup>1</sup>. Selain itu, penyimpanan nilai skripsi pun tergolong sulit karena tidak langsung dibarengi dengan nilai dan npm mahasiswa yang mengerjakan. Untuk mengatasi hal-hal tersebut, diperlukan suatu sistem yang dapat menanggulangi masalah pengisian, kalkulasi perhitungan, dan juga penyimpanan skripsi.

Menurut penjelasan di atas, maka otomatisasi sistem dalam penilaian skripsi penulis mengusulkan oleh Universitas guna mengurangi kesalahan - kesalahan kecil yang dapat berakibat fatal pada nilai mahasiswa yang bersangkutan. Berdasarkan hal tersebut dibuatlah penelitian otomatisasi sistem penilaian skripsi dengan cara membuat sebuah aplikasi berbasis web yaitu Sistem Informasi Penilaian Skripsi.

Pada penelitian ini, akan dibuat sebuah sistem penilaian yang menanggulangi masalah-masalah tersebut dengan cara membuat beberapa masukan dijadikan otomatis dan juga melakukan eksekusi perhitungan nilai akhir sesuai bobot secara otomatis. Hal ini dianggap akan memudahkan penilai dalam proses penilaian skripsi, karena penilai tidak perlu lagi repot menghitung dan juga mengisi hal-hal yang sudah terisi secara otomatis.

Dalam penelitian ini saya memakai framework AngularJS yang dimiliki oleh perusahaan *Google*. AngularJS merupakan salah satu framework yang paling sering digunakan untuk membuat sebuah aplikasi berbasis web dengan konsep *Single Page Application (SPA)*. *Single Page Application* merupakan aplikasi berbasis web yang memungkinkan sebuah halaman HTML memiliki konten - konten

---

<sup>1</sup>berdasarkan diskusi dengan dosen pembimbing

yang dapat digunakan di halaman tersebut tanpa perlu berganti ke halaman lain.

AngularJS juga bisa diintegrasikan dengan aplikasi yang menggunakan framework lain, sehingga sangat berguna dalam pengerjaan aplikasi berbasis web yang sangat luas cakupannya.

## 1.2 Rumusan Masalah

Berikut adalah susunan permasalahan yang akan dibahas pada penelitian ini:

1. Bagaimana sistem penilaian skripsi yang ada pada Program Studi Teknik Informatika di Universitas Katolik Parahyangan?
2. Bagaimana proses penyimpanan nilai skripsi?
3. Bagaimana AngularJS bekerja pada eksekusi perhitungan nilai akhir?

## 1.3 Tujuan

Berdasarkan rumusan masalah yang telah dibuat, maka tujuan penelitian ini dijelaskan ke dalam poin-poin sebagai berikut:

1. Mempelajari sistem penilaian skripsi pada Program Studi Teknik Informatika di Universitas Katolik Parahyangan
2. Merancang dan mengimplementasi proses penyimpanan nilai skripsi
3. Menentukan dan mengimplementasi AngularJS untuk mengeksekusi perhitungan nilai akhir

## 1.4 Batasan Masalah

Penelitian ini memiliki batasan-batasan seperti berikut:

1. Penelitian ini hanya dilakukan untuk form penilaian matakuliah skripsi 2

## 1.5 Metode Penelitian

Dalam penelitian ini, akan dilakukan langkah-langkah berikut:

1. Melakukan studi terhadap CodeIgniter, Twitter Bootstrap, dan AngularJS sebagai framework yang akan dipakai.
2. Melakukan perancangan untuk implementasi integrasi sistem tersebut.
3. Melakukan implementasi dari rancangan yang sudah dilakukan.
4. Melakukan pengujian pada saat sidang skripsi2 sehingga penilai dapat menguji hasil implementasi tersebut.
5. Menganalisa dan menarik kesimpulan atas hasil penelitian yang telah dilaksanakan.

---

## 1.6 Sistematika Penulisan

Berikut adalah sistematika penulisan dari dokumen ini:

- Bab 1 membahas latar belakang, rumusan masalah, tujuan penulisan, batasan-batasan, serta metode yang digunakan pada penelitian ini.
- Bab 2 membahas teori-teori yang digunakan dalam penelitian ini, yaitu AngularJS, Code Igniter, dan Twitter Bootstrap.
- Bab 3 menganalisis sistem kini, beserta perubahan-perubahan yang harus dilakukan.
- Bab 4 membahas perancangan yang dilakukan sebelum mengimplementasikan integrasi yang dimaksud, mencakup protokol, basisdata, beserta antarmukanya.
- Bab 5 membahas implementasi serta pengujian dari integrasi yang telah dilakukan.
- Bab 6 membahas kesimpulan dari keseluruhan penelitian ini, serta saran-saran yang dapat diberikan untuk penelitian berikutnya.





## BAB 2

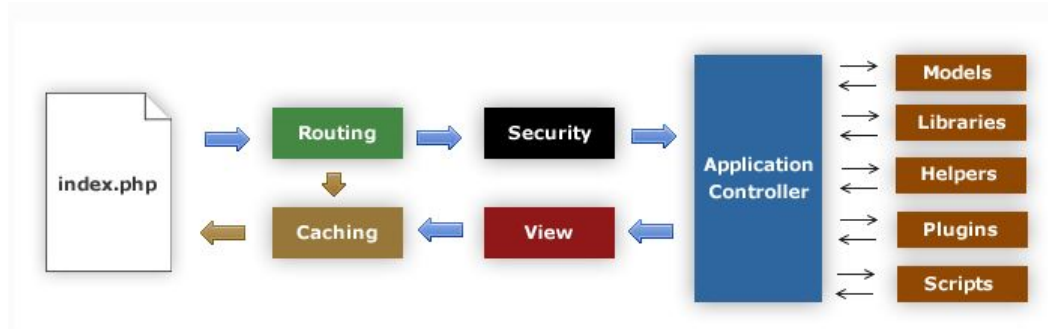
### DASAR TEORI

#### 2.1 CodeIgniter

CodeIgniter merupakan sebuah peralatan bagi orang-orang yang ingin membuat sebuah *web* dengan menggunakan bahasa PHP. CodeIgniter sendiri dibuat dengan tujuan memungkinkan pengembangan proyek-proyek lebih cepat daripada menuliskan kode dari awal. Tujuan tersebut diwujudkan dengan tersedianya *library* yang berisi *task* yang biasa dibutuhkan dalam pengembangan program dibarengi dengan antarmuka yang sederhana serta struktur logika untuk mengakses *library* tersebut. Dengan begitu dapat disimpulkan bahwa CodeIgniter membuat pemrogram fokus pada kreativitas pembuatan program dengan meminimalkan jumlah kode yang dituliskan.

##### 2.1.1 Flowchart Aplikasi CodeIgniter

Pada gambar 2.1 menunjukkan *flowchart* aliran data pada CodeIgniter:



Gambar 2.1: Flowchart CodeIgniter

Keterangan:

1. Index.php berfungsi sebagai pengontrol utama, yang menginisialisasikan sumber-sumber yang diperlukan untuk menjalankan CodeIgniter.
2. Router akan memeriksa permintaan HTTP untuk menentukan apa yang harus dilakukan selanjutnya
3. Jika terdapat *cache*, maka cache tersebut akan dikirim langsung ke browser dengan menjalankan sistem eksekusi normal.

4. HTTP *request* dan data yang diserahkan oleh *user* akan disaring oleh sistem keamanan terlebih dahulu oleh bagian keamanan(*security*) dari CodeIgniter yang dijalankan sebelum *controller* dari aplikasi diisi.
5. *Application Controller* akan mengambil isi dari *model*, *libraries*, *helpers*, *plugins*, *scripts*, dan sumber lain yang diperlukan untuk menjalankan perintah-perintah spesifik.
6. Kemudian *View* akan diterjemahkan dari *Application Controller* dan dikirim ke *web browser* untuk kemudian ditampilkan. Jika pada view final terdapat *file cache*, maka view tersebut akan terlebih dahulu dilakukan *cached* sehingga permintaan berikutnya dapat dilayani.

### 2.1.2 Model-View-Controller

CodeIgniter menggunakan dasar pola pengembangan *Model-View-Controller*(MVC). Pola pengembangan MVC ini merupakan suatu pendekatan yang memisahkan antara pengerjaan logika dan tampilan dari aplikasi.

MVC sendiri terdiri dari 3 bagian, yaitu:

1. *Model* merepresentasikan struktur data. Secara khusus, *model* merupakan kelas yang membantu menangani kueri-kueri sql seperti *insert*, *update*, dan *delete* pada basis data.
2. *View* merepresentasikan informasi yang ditunjukkan kepada pengguna. Sebuah *view* biasanya berbentuk *web page*, tetapi dalam CodeIgniter *view* bisa berbentuk *header*, *footer*, dan berbagai jenis *page* lainnya.
3. *Controller* berfungsi sebagai perantara antara *Model*, *View*, dan sumber daya lain yang diperlukan untuk memproses HTTP *request* dan menghasilkan halaman web.

## 2.2 AngularJS

AngularJS merupakan sebuah *framework* terstruktur yang digunakan untuk aplikasi web yang bersifat dinamis. Hal tersebut memungkinkan *programmer* untuk mempergunakan HTML sebagai template bahasa pemrograman dan memperluas sintaks HTML agar dapat mengekspresikan komponen aplikasi dengan jelas dan ringkas. Sifat AngularJS yang mengikat data dan mempunyai ketergantungan injeksi akan menghilangkan banyak kode yang seharusnya dituliskan oleh *programmer*, dan semua itu terjadi pada *browser* sehingga dapat disimpulkan bahwa AngularJS merupakan pasangan yang sangat ideal bagi penggunaan teknologi server. Dalam pembuatannya, ketidakcocokkan halaman statik dan dinamik biasanya diselesaikan dengan pendekatan sebagai berikut:

1. *Library*: merupakan sebuah koleksi dari berbagai macam fungsi yang berguna dalam pembuatan aplikasi *web*, contoh: JQuery.
2. *Frameworks*: merupakan suatu implementasi dari sebuah aplikasi *web* yang menempatkan kode yang dituliskan secara detail. *Framework* akan berperan melakukan pemanggilan ke kode yang dituliskan *programmer* ketika aplikasi membutuhkan sesuatu yang spesifik, contoh: durandal, ember, dll.

Dalam pembentukannya, AngularJS memiliki pendekatan yang berbeda. AngularJS berupaya untuk meminimalkan ketidakcocokan antara dokumen utama dari HTML dengan apa yang dibutuhkan oleh aplikasi untuk membuat konstruksi HTML baru. AngularJS mengajarkan *browser* sintaks baru yang disebut *directives*. Contoh contoh *directives* adalah:

1. Keterikatan data di dalam `{{}}`;
2. Dukungan untuk *Form* dan *Form Validation*
3. Pengelompokkan HTML menjadi komponen - komponen yang dapat dipakai kembali.

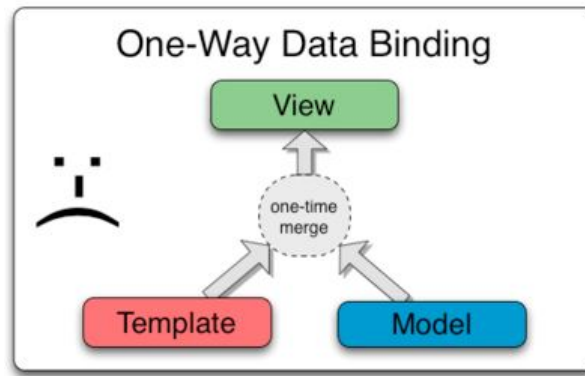
### 2.2.1 Gambaran Konseptual

Berikut ini adalah beberapa bagian-bagian terpenting dalam AngularJS.

Konsep	Deskripsi
Template	HTML dengan tambahan markup
Directives	Pengembangan HTML dengan atribut dan elemen yang dibuat khusus
Model	Data yang ditunjukkan kepada pengguna pada tampilan dan bagaimana pengguna berinteraksi
Scope	Konteks dimana model disimpan, sehingga controller, directives dan expression dapat mengaksesnya
Expression	Mengakses variabel dan fungsi dari scope
Compiler	Menguraikan template, directives, dan expression
Filter	Mengatur nilai dari sebuah expression untuk di tunjukkan kepada pengguna
View	Apa yang akan dilihat oleh pengguna (DOM)
Data Binding	Menyelaraskan data yang ada pada <i>model</i> dan view
Controller	Mengatur logika dibalik tampilan
Dependency Injection	Membuat dan menyambungkan objek dan fungsi
Injector	Tempat penyimpanan dependency Injection
Module	Tempat penyimpanan untuk bagian-bagian yang berbeda dalam sebuah aplikasi, yang mencakup: controllers, services, filters, directives yang mengkonfigurasi injector
Services	Logika bisnis independen dari views yang bisa dipakai kembali

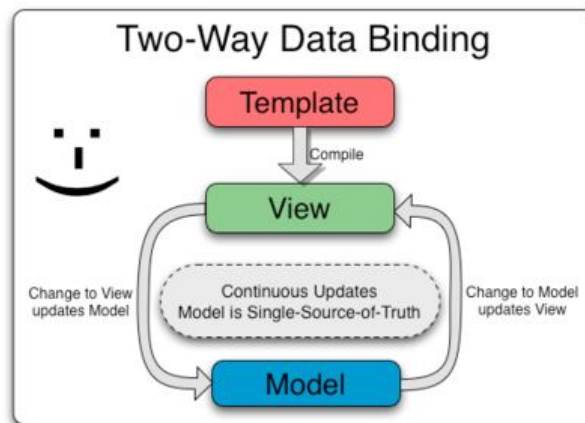
### 2.2.2 Data Binding

*Data Binding* pada AngularJS merupakan penyelarasan data antara *model* dan komponen - komponen *view*. Ketika *model* berubah, maka *view* pun akan berubah, begitu juga dengan sebaliknya.



Gambar 2.2: Data Binding Classical Templates System

Pada gambar 2.2 menjelaskan bahwa kebanyakan *data binding* adalah proses satu arah. Hal itu dilakukan dengan menyatukan *template* dan *model* menjadi *view*. Setelah penyatuan, pergantian pada *model* tidak secara otomatis mengganti *view* yang sudah ditampilkan.



Gambar 2.3: Data Binding pada Angular

Pada gambar 2.3 menjelaskan perbedaan yang diberikan oleh pelaksanaan *data binding* pada AngularJS. Pertama, *template* akan di *compile* pada browser. Hasil dari *compile* tersebut adalah *live view*. Pada tahap ini perubahan yang terjadi di *view* akan disampaikan kepada *model*, dan perubahan yang terjadi pada *model* akan mengubah *view*.

Karena *view* merupakan proyeksi dari *model*, menyebabkan *controller* benar-benar terpisahkan dari *view* tanpa disadari. Hal ini mempermudah pengujian *controller*, karena terisolasi tanpa adanya *view* dan DOM( *browser dependency*).

## 2.3 Twitter Bootstrap

## **BAB 3**

### **ANALISIS**

#### **3.1 Analisis Data Penilaian Skripsi**

Berdasarkan analisa dari contoh form penilaian skripsi yang ada, dapat disimpulkan bahwa penilaian skripsi membutuhkan data-data sebagai berikut:

- Semester
- Tahun ajaran
- NPM mahasiswa
- Nama mahasiswa
- Judul skripsi
- Nama Pembimbing utama/tunggal
- Nama Pembimbing pendamping(tidak harus)
- Nama Ketua tim penguji
- Nama Anggota tim penguji
- dan bobot masing-masing penilaian

Berdasarkan diskusi dengan dosen pembimbing, disimpulkan bahwa sistem penilaian sidang skripsi 2 ini hanya memerlukan penyimpanan untuk bobot masing-masing penilaian dan nilai akhir mahasiswa untuk tahap perhitungan. Hal ini dikarenakan nilai-nilai lainnya dapat dihasilkan dengan melakukan perhitungan pada nilai akhir mahasiswa dan bobot nilai yang diinginkan. Begitu pula dengan nilai dari masing-masing penguji.

#### **3.2 Analisis Tampilan Sistem Informasi Penilaian Skripsi**

Tampilan pada sistem informasi penilaian skripsi haruslah dibuat semirip mungkin dengan form penilaian skripsi yang sudah ada.

Perbedaan yang akan ditampilkan adalah dengan adanya otomatisasi penghitungan nilai sesuai dengan bobot yang diberikan kepada penilai. Hal ini akan memberikan kemudahan penilai untuk melakukan penilaian.

Berikut adalah bayangan awal tampilan untuk sistem informasi penilaian skripsi:

Gambar 3.1: Perbandingan Tampilan [?]

Gambar 3.2: Perkiraan Tampilan [?]

## **BAB 4**

### **PENUTUP**

#### **4.1 Kesimpulan**

Dari hasil karya ilmiah di atas, maka penulis dapat menyimpulkan bahwa penelitian membuat aplikasi sistem informasi penilaian skripsi dengan menggunakan AngularJS ini dapat mengotomatisasi perhitungan pada nilai yang diberikan oleh penilai kepada mahasiswa yang menjalankan skripsi. Dengan pengkombinasian HTML dan AngularJS, maka sistem informasi yang dihasilkan diprediksikan mempunyai tampilan yang baik dan mudah dimengerti, sekaligus mempunyai otomatisasi penghitungan dan pengisian data seperti tahun ajaran dan semester, yang dapat meminimalisir dibuatnya kelalaian penilai. Hal ini juga diprediksikan dapat memudahkan penilai dalam memberikan nilai pada mahasiswa tersebut.






## DAFTAR REFERENSI



## LAMPIRAN A

### FORM PENILAIAN SKRIPSI

Berikut adalah lembaran penilaian Skripsi yang di pakai di Program Studi Teknik Informatika Universitas Katolik Parahyangan [A.1](#):



Program Studi Teknik Informatika  
 Fakultas Teknologi Informasi dan Sains  
 Universitas Katolik Parahyangan

**Berita Acara Sidang Skripsi**  
 Semester: Ganjil/Genap\* 20..... /20.....

Telah diselenggarakan Sidang Skripsi untuk mata kuliah AIF402-6 Skripsi 2 bagi :

NPM :  Nama:

Judul :

dengan pembimbing dan penguji :

- Pembimbing Utama/Tunggal\* :
- Pembimbing Pendamping :
- Ketua Tim Penguji :
- Anggota Tim Penguji :

Rekapitulasi nilai Sidang Skripsi 2 yang diberikan oleh pembimbing, penguji & koordinator skripsi:

No	Pembimbing/Penguji	Nilai	Bobot	Nilai Akhir
1	Ketua Tim Penguji		35%	
2	Anggota Tim Penguji		35%	
3	Pembimbing		20%	
4	Koordinator Skripsi		10%	
	<b>Total</b>		<b>100%</b>	

Ditetapkan di Bandung,   20

Ketua Tim Penguji	Anggota Tim Penguji	Pembimbing**	Koordinator Skripsi

Petunjuk pengisian :

1. \* = coret yang tidak perlu, \*\* = salah satu pembimbing saja, jika pembimbing utama hadir maka harus pembimbing utama
2. Jika minimal satu penguji dan/atau seluruh pembimbing tidak hadir, maka sidang harus dibatalkan dan dijadwalkan ulang, tetapi berita acara harus tetap dilaporkan kepada koordinator skripsi atau kepada pimpinan jurusan.

Gambar A.1: Form Penilaian Skripsi saat sidang

Berikut adalah lembaran rekapitulasi penilaian Skripsi yang di pakai di Program Studi Teknik Informatika Universitas Katolik Parahyangan A.2:

**Rekapitulasi Penilaian SKRIPSI 2 (PEMBIMBING)**

NPM Mahasiswa:

Komponen Penilaian	Nilai	Bobot	Nilai Akhir
Tata Tulis Laporan		20%	
Kelengkapan Materi		20%	
Penguasaan Materi		30%	
Proses Bimbingan		30%	
		<b>Total</b>	

Tgl:    /    / 20  
 Ttd:  
 Nama:

---

**Rekapitulasi Penilaian SKRIPSI 2 (KETUA TIM PENGUJI)**

NPM Mahasiswa:

Komponen Penilaian	Nilai	Bobot	Nilai Akhir
Tata Tulis Laporan		15%	
Kelengkapan Materi		10%	
Penguasaan Materi		30%	
Presentasi		15%	
Pencapaian Tujuan		30%	
		<b>Total</b>	

Tgl:    /    / 20  
 Ttd:  
 Nama:

---

**Rekapitulasi Penilaian SKRIPSI 2 (ANGGOTA TIM PENGUJI)**

NPM Mahasiswa:

Komponen Penilaian	Nilai	Bobot	Nilai Akhir
Tata Tulis Laporan		15%	
Kelengkapan Materi		10%	
Penguasaan Materi		30%	
Presentasi		15%	
Pencapaian Tujuan		30%	
		<b>Total</b>	

Tgl:    /    / 20  
 Ttd:  
 Nama:

Gambar A.2: Form Rekapitulasi Penilaian Skripsi saat sidang

# LAMPIRAN B

## THE SOURCE CODE

Listing B.1: MyFurSet.java

```

1
2 import java.util.ArrayList;
3 import java.util.Collections;
4 import java.util.HashSet;
5
6 /**
7  *
8  * @author Lionov
9  */
10
11 //class for set of vertices close to furthest edge
12 public class MyFurSet {
13     protected int id; //id of the set
14     protected MyEdge FurthestEdge; //the furthest edge
15     protected HashSet<MyVertex> set; //set of vertices close to furthest edge
16     protected ArrayList<ArrayList<Integer>> ordered; //list of all vertices in the set for each
17         trajectory
18     protected ArrayList<Integer> closeID; //store the ID of all vertices
19     protected ArrayList<Double> closeDist; //store the distance of all vertices
20     protected int totaltrj; //total trajectories in the set
21
22     /**
23      * Constructor
24      * @param id : id of the set
25      * @param totaltrj : total number of trajectories in the set
26      * @param FurthestEdge : the furthest edge
27      */
28     public MyFurSet(int id,int totaltrj,MyEdge FurthestEdge) {
29         this.id = id;
30         this.totaltrj = totaltrj;
31         this.FurthestEdge = FurthestEdge;
32         set = new HashSet<MyVertex>();
33         ordered = new ArrayList<ArrayList<Integer>>();
34         for (int i=0;i<totaltrj;i++) ordered.add(new ArrayList<Integer>());
35         closeID = new ArrayList<Integer>(totaltrj);
36         closeDist = new ArrayList<Double>(totaltrj);
37         for (int i = 0; i < totaltrj; i++) {
38             closeID.add(-1);
39             closeDist.add(Double.MAX_VALUE);
40         }
41     }
42
43     /**
44      * set a vertex into the set
45      * @param v : vertex to be added to the set
46      */
47     public void add(MyVertex v) {
48         set.add(v);
49     }
50
51     /**
52      * check whether vertex v is a member of the set
53      * @param v : vertex to be checked
54      * @return true if v is a member of the set , false otherwise
55      */
56     public boolean contains(MyVertex v) {
57         return this.set.contains(v);
58     }
59 }

```