«SKRIPSI/TUGAS AKHIR»

KONVERSI SHARIF JUDGE DARI CODEIGNITER 3 KE CODEIGNITER 4

Filipus

NPM: 6181901074

PROGRAM STUDI «MATEMATIKA/FISIKA/TEKNIK INFORMATIKA»
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN

«tahun»

«FINAL PROJECT/UNDERGRADUATE THESIS»

«JUDUL BAHASA INGGRIS»

Filipus

NPM: 6181901074

DEPARTMENT OF «MATHEMATICS/PHYSICS/INFORMATICS»
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES
PARAHYANGAN CATHOLIC UNIVERSITY
«tahun»

LEMBAR PENGESAHAN

KONVERSI SHARIF JUDGE DARI CODEIGNITER 3 KE CODEIGNITER 4

Filipus

NPM: 6181901074

Bandung, «tanggal» «bulan» «tahun»

Menyetujui,

Pembimbing Utama	Pembimbing Pendamping
«pembimbing utama/1»	«pembimbing pendamping/2»

Ketua Tim Penguji	Anggota Tim Penguji
«penguji 1»	«penguji 2»

Mengetahui,

Ketua Program Studi

«Ketua Program Studi»

PERNYATAAN

Dengan ini saya yang bertandatangan di bawah ini menyatakan bahwa «skripsi/tugas akhir» dengan judul:

KONVERSI SHARIF JUDGE DARI CODEIGNITER 3 KE CODEIGNITER 4

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»

Filipus
NPM: 6181901074

ABSTRAK

«Tuliskan abstrak anda di sini, dalam bahasa Indonesia»

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

ABSTRACT

«Tuliskan abstrak anda di sini, dalam bahasa Inggris»

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

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

KATA PENGANTAR

«Tuliskan kata pengantar dari anda di sini . . . »

Bandung, «bulan» «tahun»

Penulis

DAFTAR ISI

Kata Pengantar xv

Daftar Isi xvii

Daftar Gambar xix

1 Pendahuluan 1
- 1.1 Latar Belakang . . . 1
- 1.2 Rumusan Masalah . . . 2
- 1.3 Tujuan . . . 2
- 1.4 Batasan Masalah . . . 2
- 1.5 Metodologi . . . 2
- 1.6 Sistematika Pembahasan . . . 2

2 Landasan Teori 5
- 2.1 SharIF Judge . . . 5
  - 2.1.1 Instalasi . . . 5
  - 2.1.2 *Clean Urls* . . . 6
  - 2.1.3 Users . . . 6
  - 2.1.4 Menambah Assignment . . . 6
  - 2.1.5 Sample Assignment . . . 6
- 2.2 CodeIgniter 3 . . . 6
- 2.3 CodeIgniter 4 . . . 6

6section.2.4
  - 2.4.1 Struktur Aplikasi . . . 6
  - 2.4.2 Tabel . . . 8
  - 2.4.3 Kutipan . . . 8
  - 2.4.4 Gambar . . . 9
  - 2.4.5 Kode Program . . . 11
  - 2.4.6 Notasi . . . 11

A Kode Program 13

B Hasil Eksperimen 15

DAFTAR GAMBAR

1.1 Tampilan halaman *SharIF Judge* 1

2.1 Gambar arsitektur MVC 6
2.2 Gambar *Serpentes* dalam format png 10
2.3 Ular kecil 10
2.4 *Serpentes* betina 11

B.1 Hasil 1 15
B.2 Hasil 2 15
B.3 Hasil 3 15
B.4 Hasil 4 15

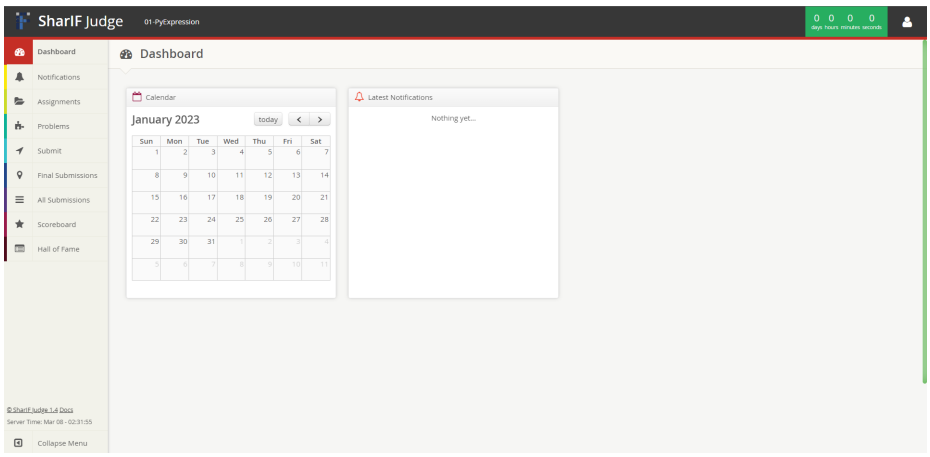
BAB 1

PENDAHULUAN

1.1 Latar Belakang

Tugas merupakan suatu bentuk pembelajaran dan penilaian yang diberikan oleh pengajar kepada pelajar untuk membantu pelajar mendalami materi yang sudah diberikan. Pembagian tugas yang diberikan dapat dibagi menjadi 2 jenis yakni tugas individu dan tugas kelompok. Tugas individu merupakan tugas yang hanya ditanggung oleh satu individu sedangkan, tugas kelompok merupakan tugas yang ditanggung oleh beberapa individu. Tugas selanjutnya akan dikumpulkan kepada pengajar dan diberikan penilaian berdasarkan tingkat ketepatan jawaban dari tugas tersebut. Pengumpulan dan pengecekan tugas terutama *coding* secara manual memiliki kekurangan dimana diperlukan banyak langkah dalam melakukan pengecekan dan pengiriman nilai. Pengecekan secara manual juga terdapat kesulitan dalam pengecekan yakni, kekurangan dalam pengecekan plagiat antara tugas pelajar. Maka, dibutuhkan perangkat lunak untuk melakukan pengecekan secara otomatis salah satunya adalah *Online Judge*.

Online Judge merupakan sebuah perangkat lunak berbasis web yang dapat melakukan pengecekan *program* sesuai dengan standar yang sudah diberikan. Perangkat lunak ini dapat menerima jawaban dari pelajar dan melakukan pengecekan secara otomatis dan memberikan keluaran berupa nilai dari pelajar tersebut. Salah satu perangkat lunak *Online Judge* terdapat pada Informatika Unpar bernama SharIF Judge (dapat dilihat pada Gambar 1.1).



Gambar 1.1: Tampilan halaman *SharIF Judge*

SharIF Judge merupakan sebuah alat *open source* untuk menilai kode dengan beberapa bahasa seperti C, C++, Java, dan Python secara online. SharIF Judge dibentuk menggunakan *framework* CodeIgniter 3 yang merupakan *framework* berbasis PHP dan dimodifikasi sesuai dengan kebutuhan Informatika Unpar untuk mengumpulkan tugas dan ujian mahasiswa.

CodeIgniter 3 merupakan sebuah *framework* gratis yang bertujuan untuk mempermudah dalam membentuk sebuah aplikasi *website* menggunakan PHP. CodeIgniter 3 menggunakan struktur

MVC yang membagi file menjadi 3 buah yaitu Model, View, Controller. Selain itu, CodeIgniter 3 merupakan *framework* ringan dan menyediakan banyak *library* untuk digunakan oleh penggunanya.

CodeIgniter 3 sudah memasuki fase *maintenance* sehingga tidak akan mendapatkan *update* lebih lanjut dari pembentuknya. CodeIgniter 3 pada akhirnya akan tidak dapat dipakai dan akan hilangnya dokumentasi dari situs web resmi. Sehingga, perangkat lunak yang menggunakan CodeIgniter 3 perlu dikonversi ke *framework* CodeIgniter dengan versi terbaru yakni CodeIgniter 4.

CodeIgniter 4 merupakan versi terbaru dari *framework* CodeIgniter yang memiliki banyak perubahan fitur dari versi sebelumnya. CodeIgniter 4 dibentuk menggunakan versi PHP 7.4 sedangkan CodeIgniter 3 dibentuk menggunakan versi PHP 5.6. CodeIgniter 4 membagi file menggunakan struktur MVC namun, memiliki struktrur folder berbeda dengan versi sebelumnya.

Pada skripsi ini, akan dilakukan konversi SharIF Judge dari CodeIgniter 3 menjadi CodeIgniter 4. Konversi dilakukan karena CodeIgniter 3 sudah memasukin fase *maintenance* sehingga CodeIgniter 3 hanya akan mendapatkan *security update*.

1.2 Rumusan Masalah

- Apa standar yang ada sehingga CodeIgniter 3 perlu dikoversi CodeIgniter 4?
- Bagaimana cara melakukan konversi CodeIgniter 3 menjadi CodeIgniter 4?
- Bagaimana mengevaluasi kode SharIF Judge dan mengubahnya agar dapat berjalan di CodeIgniter 4?

1.3 Tujuan

- Mencari standar yang dibutuhkan sehingga CodeIgniter 3 perlu di konversi mendjadi CodeIgniter 4.
- Melakukan konversi dengan megubah kode sesuai dengan CodeIgniter 4.
- Melakukan evaluasi kode SharIF Judge dan mengubahnya agar dapat berjalan di CodeIgniter 4.

1.4 Batasan Masalah

Perangkat lunak hanya akan dikonversi sesuai dengan perangkat lunak yang sudah dibentuk. Kesalahan-kesalahan lain (tidak dapat di run pada windows,) akan ditulis di keluaran perangkat lunak apa adanya.

1.5 Metodologi

Metodologi yang dilakukan dalam melakukan penelitian ini adalah sebagian berikut:

1. Melakukan analisis dan eksplorasi fungsi-fungsi perangkat lunak SharIF Judge.
2. Melakukan analisis
3. Melakukan konversi perangkat lunak dari CodeIgniter 3 menjadi CodeIgniter 4.
4. Melakukan pengujian dan eksperimen terhadap perangkat lunak yang sudah di konversi.
5. Menyelesaikan pembentukan dokumen

1.6 Sistematika Pembahasan

Penelitian ini akan dibahas dalam enam bab yang masing-masing berisi:

1. **Bab 1:** Pendahuluan Bab ini berisi latar belakang, rumusan masalah,tujuan, batasan masalah, metodologi, dan sistematika pembahasan.
2. **Bab 2:** Dasar Teori Bab ini berisi

3. **Bab 3:** Analisis
4. **Bab 4:** Perancangan
5. **Bab 5:** Implementasi dan Pengujian
6. **Bab 6:** Kesimpulan dan Saran

BAB 2

LANDASAN TEORI

2.1 SharIF Judge

SharIF Judge merupakan sebuah *Online Judge* yang diambil dari Mohammed Javad Naderi yang dibentuk menggunakan CodeIgniter 3 dan diubah sesuai dengan kebutuhan di Informatika Universitas Katolik Parahyangan. SharIF Judge dapat menilai kode berbahasa *C, C++, Java, dan Python* dengan mengunggah file ataupun mengetiknya langsung.

2.1.1 Instalasi

Berikut merupakan persyaratan dan langkah-langkah melakukan *instalasi SharIF Judge*:

Persyaratan

SharIF Judge dapat dijalankan pada sistem operasi *Linux* dengan syarat sebagai berikut:

- Diperlukan *webserver* dengan versi PHP 5.3 atau lebih baru.
- Pengguna dapat menjalankan PHP pada *command line*. Pada *Ubuntu* diperlukan instalasi paket *php5-cli*.
- *MySql database* dengan ekstensi *Mysqli* untuk PHP atau *PostgreSql database*.
- PHP harus memiliki akses untuk menjalankan perintah melalui fungsi *shell_exec*.

Kode 2.1: Kode untuk melakukah pengetesan fungsi $shell_exec$

```
1    echo shell_exec("php -v");
```

- *Tools* untuk melakukan kompilasi dan menjalankan kode yang dikumpulkan (*gcc, g++, javac, java, python2, python3*).
- *Perl* disarankan untuk diinstalasi untuk alasan ketepatan waktu, batas memori, dan memaksimalkan batas ukuran pada hasil kode yang dikirim.

Instalasi

- Mengunduh versi terakhir dari *SharIF Judge* dan melakukan *unpack* pada direktori *public html*.
- Memindahkan *folder system* dan *application* diluar direktori *public* dan mengubah *path* pada `index.php`(Opsional).

Kode 2.2: Contoh *path* pada halaman index.php

```
1    $system_path = '/home/mohammad/secret/system';
2    application_folder = '/home/mohammad/secret/application';
```

- Membentuk *database MySql* atau *PostgreSql* untuk *SharIF Judge*. Jangan melakukan instalasi paket koneksi *database* apapun untuk *C, C++, Java,* atau *Python*.

2.1.2 *Clean Urls*

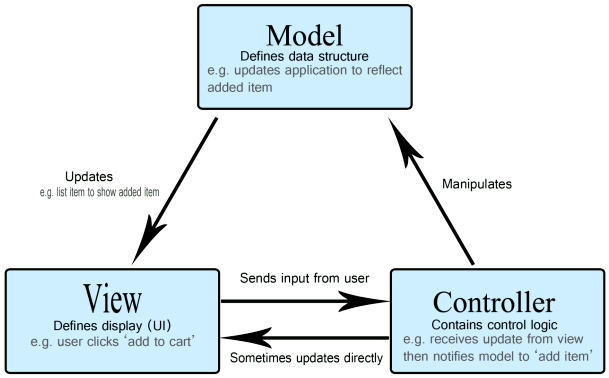
2.1.3 Users

2.1.4 Menambah Assignment

2.1.5 Sample Assignment

2.2 CodeIgniter 3

CodeIgniter 3 merupakan sebuah *framework* yang berfungsi untuk mempermudah penggunanya dalam membentuk aplikasi *website* menggunakan bahasa PHP. CodeIgniter 3 menggunakan struktur MVC (dapat dilihat pada Gambar 2.1)) dalam pembentukan *website*.



Gambar 2.1: Gambar arsitektur MVC

MVC membagi struktur *folder* menjadi 3 buah dengan fungsi berbeda yaitu Model, View, Controller. Model

2.3 CodeIgniter 4

CodeIgniter 4 merupakan versi terbaru dari *framework* CodeIgniter.

2.4 Koversi CodeIgniter 3 ke CodeIgniter 4 [1]

CodeIgniter 3 sedang mengalami fase *maintenance* sehingga perlu dilakukan konversi menjadi CodeIgniter 4. Konversi CodeIgniter 3 ke CodeIgniter 4 diperlukan penulisan ulang karena terdapat banyak implementasi yang berbeda. Konversi ke CodeIgniter 4 diawali dengan melakukan instalasi projek baru CodeIgniter 4.

2.4.1 Struktur Aplikasi

Struktur direktori pada CodeIgniter 4 memiliki perubahan yang terdiri *app, public, dan writable.* Direktori app merupakan perubahan dari direktori application dengan isi yang hampir sama dengan beberapa perubahan nama dan perpindahan direktori. Pada CodeIgniter 4 terdapat direktori *public* yang bertujuan sebagai direktori utama pada aplikasi website. Selanjutnya terdapat direktori *writable* yang berisikan *cache data, logs,* dan *session data.*

[1] https://codeigniter.com/user_guide/installation/upgrade_4xx.html

Routing

CodeIgniter 4 meletakan *route* pada *file* `app\Config\Routes.php`. CodeIgniter 4 memiliki fitur *auto routing* seperti pada CodeIgniter 3 namun, pada *default* di matikan. Fitur *auto routing* memungkinkan untuk dinyalakan serupa dengan pada CodeIgniter 3 namun tidak direkomendasikan karena alasan *security*.

Model, View, and Controller

Struktur MVC pada CodeIgniter 4 berbeda dibandingkan CodeIgniter 3 dimana pada CodeIgniter 4 direktori *Model* terdapat pada direktori `app\Models`. Pembentukan *file* untuk *Model* perlu ditambahkan `namespace App\Models;` dan `use CodeIgniter\Model;` pada awal *file* setelah membuka tag PHP. Selanjutnya nama fungsi perlu diubah dari `extends CI_Model` menjadi `extends Model`.

View pada CodeIgniter 4 terdapat di `app\Views` dengan sintaks yang harus diubah. Sintaks yang harus diubah merupakan sintaks untuk memanggil *view* pada CodeIgniter 3 `$this->load->view('x');` sedangkan pada CodeIgniter 4 dapat menggunakan `return view('x);`. Selanjutnya, sintaks `<?php echo $title?>` pada halaman *view* dapat diubah menjadi `<?= $title ?>`.

Controller pada CodeIgniter 4 terdapat di `app\Controllers` dan diperlukan beberapa perubahan. Pertama, perlu ditambahkan `namespace App\Controllers;` pada awal *file* setelah membuka tag PHP. Selanjutnya, perlu mengubah `extends CI_Controller` menjadi `extends BaseController`. Selanjutnya, diperlukan pengubahan nama pada pemanggilan *file* menjadi `App\Controllers\User.php` .

Libraries

CodeIgniter 4 menyediakan *library* untuk digunakan dan dapat diinstall apabila diperlukan. Pemanggilan *library* berubah dari `$this->load->library('x');` menjadi `$this->x = new X();`. Terdapat beberapa *library* yang harus di *upgrade* dengan sedikit perubahan.

Helpers

Helpers tidak terdapat banyak perubahan namun, beberapa *helpers* pada *CodeIgniter 3* tidak terdapat pada *CodeIgniter 4* sehingga perlu perubahan pada implementasi fungsinya. *Helpers* dapat di dimuat secara otomatis menggunakan `app\Config\Autoload.php`

Events

Framework

Configuration

File configuration CodeIgniter 4 terdapat pada `app\Config` dengan penulisan sedikit berbeda dengan versi sebelumnya. Pengguna hanya perlu melakukan pemindahan menuju CodeIgniter 4 dan apabila menggunakan *file config custom* maka, diperlukan penulisan ulang pada direktori Config dengan melakukan *extend* pada `CodeIgniter\Config\BaseConfig`.

Database

Penggunaan *database* pada CodeIgniter 4 hanya berubah sedikit dibandingkan dengan versi sebelumnya. Data-data penting kredensial diletakan pada `app\Config\Database.php` dan perlu dilakukan beberapa perubahan sintaks dan *query*. Sintaks untuk memuat database diubah menjadi `$db = db_connect();` dan apabila menggunakan beberapa *database* maka sintaks menjadi `$db = db_connect('group_name');`. Semua *query* harus diubah dari `$this->db` menjadi `$db` dan beberapa sintaks perlu diubah seperti `$query->result();` menjadi `$query->getResult();`. Selain itu, terdapat *class* baru yakni *Query Builder Class* yang harus di inisiasi `$builder = $db->table('mytable');` dan dapat dipakai untuk menjalankan *query* dengan mengganti `$this->db` seperti `$this->db->get();` menjadi `$builder->get();`.

Emails

Perubahan *email* hanya terdapat pada nama dari *method* dan pemanggilan *library email*. Pemanggilan *library* berubah dari `$this->load->library('email);` menjadi `$email = service('email');` dan selanjutnya perlu dilakukan perubahan pada semua `$this->email` menjadi `$email`. Selanjutnya beberapa pemanggilan *method* berubah dengan tambahan *set* didepannya seperti *from* menjadi *setFrom*.

Encryption

Terdapat perubahan

Working with Uploaded Files

Terdapat banyak perubahan dimana pada CodeIgniter 4 pengguna dapat mengecek apakah *file* telah terunggah tanpa *error* dan lebih mudah untuk melakukan penyimpanan *file*. Pada CodeIgniter 4 melakukan akses pada *uploaded file* dilakukan dengan `$file = $this->request->getFile('userfile')` dan dapat dilakukan validasi dengan cara `$file->isValid()`. Penyimpanan *file* yang sudah diunggah dapat dilakukan dengan `$path = $this->request->getFile('userfile')->store('head_img/', 'user_n` *File* yang sudah diunggah dan di validasi akan tersimpan pada `writable/uploads/head_img/user_name.jpg`.

HTML Tables

Tidak terdapat banyak perubahan pada *framework* versi terbaru hanya perubahan pada nama *method* dan pemanggilan *library*. Perubahan pemanggilan *library* dari `$this->load->library('table');` menjadi `$table = new \CodeIgniter\View\Table();` dan perlu dilakukan perubahan setiap `$this->table` menjadi `$table`. Selain itu, terdapat bebera perubahan pada penamaan *method* dari *underscored* menjadi *camelCase*.

Localization

CodeIgniter 4 mengembalikan *file* bahasa menjadi *array* sehingga perlu dilakukan beberapa perubahan. Pertama, perlu dilakukan konfigurasi *default languange* pada perangkat lunak. Selanjutnya melakukan pemindahan *file* bahasa pada *CodeIgniter 3* menuju `app\Languange\<locale>`. *File-file* bahasa *CodeIgniter 3* perlu dilakukan penghapusan semua kode `$this->lang->load($file, $lang);` dan mengubah *method* pemanggilan bahasa dari `$this->lang->line('error_email_missing')` menjadi `echo lang('Errors.errorEmailMissing');`

Migrations

Perubahan perlu dilakukan pada nama *file* menjadi nama dengan cap waktu. Selanjutnya dilakukan penghapusan kode `defined('BASEPATH') OR exit('No direct script access allowed');` dan menambahkan dua buah kode yakni, `namespace App\Database\Migrations;` dan `use CodeIgniter\Databas` setelah membuka tag PHP. Setelah itu, `extends CI_Migration` diubah menjadi `extends Migration`. Terakhir, terdapat perubahan pada nama *method Forge* dari `$this->dbforge->add_field` menjadi *camelCase* `$this->forge->addField`.

Pagination

HTTP Response

Routing

Security

Sessions

Validations

View Parser

2.4.2 Tabel

Berikut adalah contoh pembuatan tabel. Penempatan tabel dan gambar secara umum diatur secara otomatis oleh LaTeX, perhatikan contoh di file bab2.tex untuk melihat bagaimana cara memaksa tabel ditempatkan sesuai keinginan kita.

Perhatikan bawa berbeda dengan penempatan judul gambar gambar, keterangan tabel harus diletakkan di atas tabel!! Lihat Tabel 2.1 berikut ini:

Tabel 2.1: Tabel contoh

	v_{start}	$\mathcal{S}_1$	v_{end}
τ_1	1	12	20
τ_2	1		20
τ_3	1	9	20
τ_4	1		20

Tabel 2.2 dan Tabel 2.3 berikut ini adalah tabel dengan sel yang berwarna dan ada dua tabel yang bersebelahan.

Tabel 2.2: Tabel bewarna(1)

	v_{start}	$\mathcal{S}_2$	$\mathcal{S}_1$	v_{end}
τ_1	1	5	12	20
τ_2	1	8		20
τ_3	1	2/8/17	9	20
τ_4	1			20

Tabel 2.3: Tabel bewarna(2)

	v_{start}	$\mathcal{S}_1$	$\mathcal{S}_2$	v_{end}
τ_1	1	12	5	20
τ_2	1		8	20
τ_3	1	9	2/8/17	20
τ_4	1			20

2.4.3 Kutipan

Berikut contoh kutipan dari berbagai sumber, untuk keterangan lebih lengkap, silahkan membaca file referensi.bib yang disediakan juga di template ini. Contoh kutipan:

- Buku: [?]
- Bab dalam buku: [?]
- Artikel dari Jurnal: [?]
- Artikel dari prosiding seminar/konferensi: [?]
- Skripsi/Thesis/Disertasi: [?] [?] [?]
- Technical/Scientific Report: [?]
- RFC (Request For Comments): [?]
- Technical Documentation/Technical Manual: [?] [?] [?]
- Paten: [?]
- Tidak dipublikasikan: [?] [?]
- Laman web: [?]
- Lain-lain: [?]

2.4.4 Gambar

Pada hampir semua editor, penempatan gambar di dalam dokumen LaTeX tidak dapat dilakukan melalui proses *drag and drop*. Perhatikan contoh pada file bab2.tex untuk melihat bagaimana cara menempatkan gambar. Beberapa hal yang harus diperhatikan pada saat menempatkan gambar:

- Setiap gambar **harus** diacu di dalam teks (gunakan *field* LABEL)
- *Field* CAPTION digunakan untuk teks pengantar pada gambar. Terdapat dua bagian yaitu yang ada di antara tanda [dan] dan yang ada di antara tanda { dan }. Yang pertama akan muncul di Daftar Gambar, sedangkan yang kedua akan muncul di teks pengantar gambar. Untuk skripsi ini, samakan isi keduanya.
- Jenis file yang dapat digunakan sebagai gambar cukup banyak, tetapi yang paling populer adalah tipe PNG (lihat Gambar 2.2), tipe JPG (Gambar 2.3) dan tipe PDF (Gambar 2.4)
- Besarnya gambar dapat diatur dengan *field* SCALE.
- Penempatan gambar diatur menggunakan *placement specifier* (di antara tanda [dan] setelah deklarasi gambar. Yang umum digunakan adalah **H** untuk menempatkan gambar **sesuai** penempatannya di file .tex atau **h** yang berarti "kira-kira" di sini.
  Jika tidak menggunakan *placement specifier*, LaTeX akan menempatkan gambar secara otomatis untuk menghindari bagian kosong pada dokumen anda. Walaupun cara ini sangat mudah, hindarkan terjadinya penempatan dua gambar secara berurutan.
  - Gambar 2.2 ditempatkan di bagian atas halaman, walaupun penempatannya dilakukan setelah penulisan 3 paragraf setelah penjelasan ini.
  - Gambar 2.3 dengan skala 0.5 ditempatkan di antara dua buah paragraf. Perhatikan penulisannya di dalam file bab2.tex!
  - Gambar 2.4 ditempatkan menggunakan *specifier* **h**.

Gambar 2.2: Gambar *Serpentes* dalam format png

Curabitur tellus magna, porttitor a, commodo a, commodo in, tortor. Donec interdum. Praesent scelerisque. Maecenas posuere sodales odio. Vivamus metus lacus, varius quis, imperdiet quis, rhoncus a, turpis. Etiam ligula arcu, elementum a, venenatis quis, sollicitudin sed, metus. Donec nunc pede, tincidunt in, venenatis vitae, faucibus vel, nibh. Pellentesque wisi. Nullam malesuada. Morbi ut tellus ut pede tincidunt porta. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam congue neque id dolor.

Donec et nisl at wisi luctus bibendum. Nam interdum tellus ac libero. Sed sem justo, laoreet vitae, fringilla at, adipiscing ut, nibh. Maecenas non sem quis tortor eleifend fermentum. Etiam id tortor ac mauris porta vulputate. Integer porta neque vitae massa. Maecenas tempus libero a libero posuere dictum. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Aenean quis mauris sed elit commodo placerat. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Vivamus rhoncus tincidunt libero. Etiam elementum pretium justo. Vivamus est. Morbi a tellus eget pede tristique commodo. Nulla nisl. Vestibulum sed nisl eu sapien cursus rutrum.

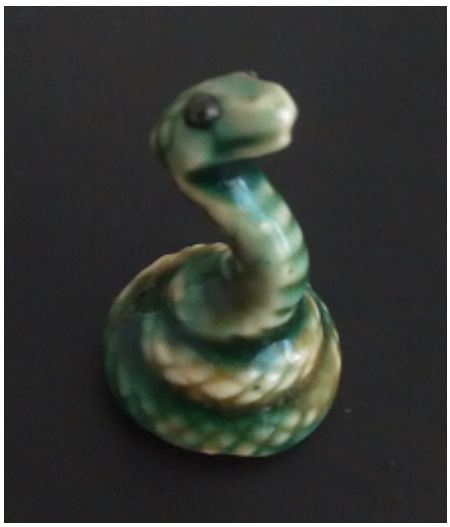
Nulla non mauris vitae wisi posuere convallis. Sed eu nulla nec eros scelerisque pharetra. Nullam varius. Etiam dignissim elementum metus. Vestibulum faucibus, metus sit amet mattis rhoncus, sapien dui laoreet odio, nec ultricies nibh augue a enim. Fusce in ligula. Quisque at magna et nulla commodo consequat. Proin accumsan imperdiet sem. Nunc porta. Donec feugiat mi at justo. Phasellus facilisis ipsum quis ante. In ac elit eget ipsum pharetra faucibus. Maecenas viverra nulla in massa.

Nulla ac nisl. Nullam urna nulla, ullamcorper in, interdum sit amet, gravida ut, risus. Aenean ac enim. In luctus. Phasellus eu quam vitae turpis viverra pellentesque. Duis feugiat felis ut enim. Phasellus pharetra, sem id porttitor sodales, magna nunc aliquet nibh, nec blandit nisl mauris at pede. Suspendisse risus risus, lobortis eget, semper at, imperdiet sit amet, quam. Quisque scelerisque dapibus nibh. Nam enim. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nunc ut metus. Ut metus justo, auctor at, ultrices eu, sagittis ut, purus. Aliquam aliquam.

Gambar 2.3: Ular kecil

Etiam pede massa, dapibus vitae, rhoncus in, placerat posuere, odio. Vestibulum luctus commodo lacus. Morbi lacus dui, tempor sed, euismod eget, condimentum at, tortor. Phasellus aliquet odio ac lacus tempor faucibus. Praesent sed sem. Praesent iaculis. Cras rhoncus tellus sed justo ullamcorper sagittis. Donec quis orci. Sed ut tortor quis tellus euismod tincidunt. Suspendisse congue nisl eu elit. Aliquam tortor diam, tempus id, tristique eget, sodales vel, nulla. Praesent tellus mi, condimentum sed, viverra at, consectetuer quis, lectus. In auctor vehicula orci. Sed pede sapien, euismod in, suscipit in, pharetra placerat, metus. Vivamus commodo dui non odio. Donec et felis.

Etiam suscipit aliquam arcu. Aliquam sit amet est ac purus bibendum congue. Sed in eros. Morbi non orci. Pellentesque mattis lacinia elit. Fusce molestie velit in ligula. Nullam et orci vitae nibh vulputate auctor. Aliquam eget purus. Nulla auctor wisi sed ipsum. Morbi porttitor tellus ac enim. Fusce ornare. Proin ipsum enim, tincidunt in, ornare venenatis, molestie a, augue. Donec vel pede in lacus sagittis porta. Sed hendrerit ipsum quis nisl. Suspendisse quis massa ac nibh pretium cursus. Sed sodales. Nam eu neque quis pede dignissim ornare. Maecenas eu purus ac urna tincidunt congue.



Gambar 2.4: *Serpentes* jantan

2.4.5 Kode Program

Kode program dalam bahasa tertentu seringkali harus ditulis di dalam bab, bukan hanya dilampirkan di bagian Lampiran. Kode 2.3 menampilkan penggunaan karakter-karakter yang umum digunakan dalam sebuah program yang ditulis dengan bahasa C.

Kode 2.3: Kode untuk menampilkan karakter-karakter aneh

```
1 // This does not make algorithmic sense,
2 // but it shows off significant programming characters.
3
4 #include<stdio.h>
```

```
5
6  void myFunction( int input, float* output ) {
7      switch ( array[i] ) {
8          case 1: // This is silly code
9              if ( a >= 0 || b <= 3 && c != x )
10                 *output += 0.005 + 20050;
11             char = 'g';
12             b = 2^n + ~right_size - leftSize * MAX_SIZE;
13             c = (--aaa + &daa) / (bbb++ - ccc % 2 );
14             strcpy(a,"hello_$@?");
15     }
16     count = ~mask | 0x00FF00AA;
17 }
18
19 // Fonts for Displaying Program Code in LATEX
20 // Adrian P. Robson, nepsweb.co.uk
21 // 8 October 2012
22 // http://nepsweb.co.uk/docs/progfonts.pdf
```

2.4.6 Notasi

Simbol-simbol (matematika) yang sering digunakan sepanjang penulisan skripsi, dapat dimasukkan ke dalam "Daftar Notasi". Daftar ini ada di halaman depan sebelum Bab 1. Cara memasukkan sebuah simbol ke dalam Daftar Notasi adalah menggunakan perintah `\nomenclature`. Contoh:

```
\nomenclature[]{$A$}{luas kandang ular}
```

Argumen opsional digunakan untuk mengurutkan notasi. Silahkan lihat sendiri dokumentasi package `nomencl`

LAMPIRAN A

KODE PROGRAM

Kode A.1: MyCode.c

```c

// This does not make algorithmic sense,
// but it shows off significant programming characters.

#include<stdio.h>

void myFunction( int input, float* output ) {
    switch ( array[i] ) {
        case 1: // This is silly code
            if ( a >= 0 || b <= 3 && c != x )
                *output += 0.005 + 20050;
            char = 'g';
            b = 2^n + ~right_size - leftSize * MAX_SIZE;
            c = (--aaa + &daa) / (bbb++ - ccc % 2 );
            strcpy(a,"hello $@?");
    }
    count = ~mask | 0x00FF00AA;
}

// Fonts for Displaying Program Code in LATEX
// Adrian P. Robson, nepsweb.co.uk
// 8 October 2012
// http://nepsweb.co.uk/docs/progfonts.pdf
```

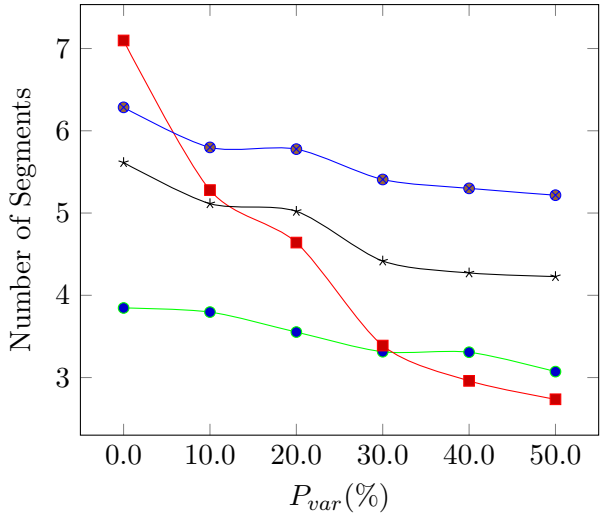
Kode A.2: MyCode.java

```java
import java.util.ArrayList;
import java.util.Collections;
import java.util.HashSet;

//class for set of vertices close to furthest edge
public class MyFurSet {
    protected int id;                                   //id of the set
    protected MyEdge FurthestEdge;                      //the furthest edge
    protected HashSet<MyVertex> set;                    //set of vertices close to furthest edge
    protected ArrayList<ArrayList<Integer>> ordered;    //list of all vertices in the set for each trajectory
    protected ArrayList<Integer> closeID;               //store the ID of all vertices
    protected ArrayList<Double> closeDist;              //store the distance of all vertices
    protected int totaltrj;                             //total trajectories in the set

    /*
     * Constructor
     * @param id : id of the set
     * @param totaltrj : total number of trajectories in the set
     * @param FurthestEdge : the furthest edge
     */
    public MyFurSet(int id,int totaltrj,MyEdge FurthestEdge) {
        this.id = id;
        this.totaltrj = totaltrj;
        this.FurthestEdge = FurthestEdge;
        set = new HashSet<MyVertex>();
        ordered = new ArrayList<ArrayList<Integer>>();
        for (int i=0;i<totaltrj;i++) ordered.add(new ArrayList<Integer>());
        closeID = new ArrayList<Integer>(totaltrj);
        closeDist = new ArrayList<Double>(totaltrj);
        for (int i = 0;i <totaltrj;i++) {
            closeID.add(-1);
            closeDist.add(Double.MAX_VALUE);
        }
    }

}
```

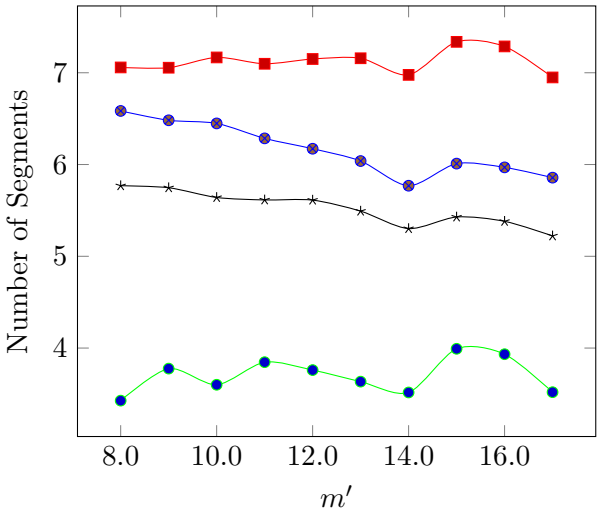
LAMPIRAN B

HASIL EKSPERIMEN

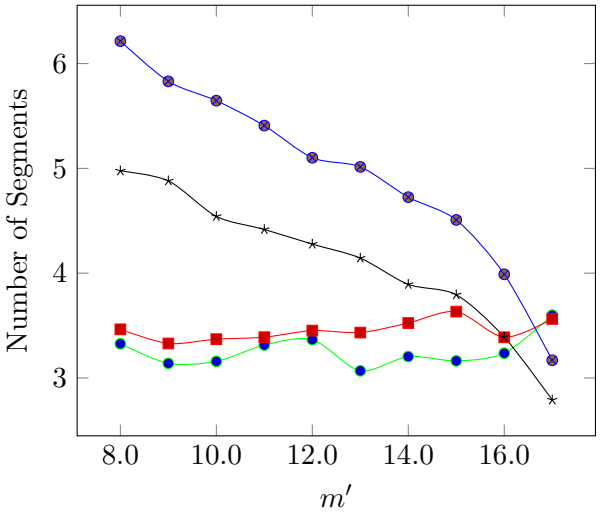
Hasil eksperimen berikut dibuat dengan menggunakan TIKZPICTURE (bukan hasil excel yg diubah ke file bitmap). Sangat berguna jika ingin menampilkan tabel (yang kuantitasnya sangat banyak) yang datanya dihasilkan dari program komputer.



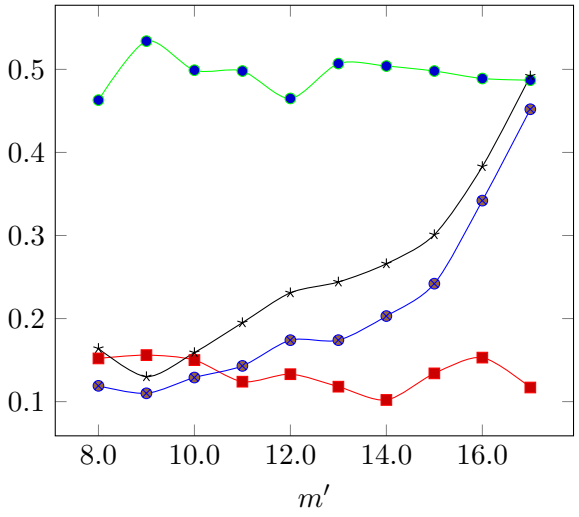
Gambar B.1: Hasil 1



Gambar B.2: Hasil 2



Gambar B.3: Hasil 3



Gambar B.4: Hasil 4