

# Ketentuan Update CMS Master:

1. Kebutuhan client tidak sesuai dengan kondisi CMS sekarang
2. Versi codeigniter CMS atau versi PHP sudah lawas
3. Ada isu terkait *security*, *bug* terhadap fungsi yang sudah ada
4. Ada update fungsi terbaru dari *third-party*
5. Ada fitur baru, *library* baru yang bisa digunakan secara *general* (tidak hanya berkaitan dengan satu projek)

# Efisiensi Penulisan Program:

1. Gunakan ***sublime*** sebagai file editor dan pelajari shortcut-shortcut / quick key nya untuk mempermudah, mempercepat, serta efisiensi waktu
2. Selalu gunakan fungsi dan library yang sudah ada
3. Bedakan css/js *general* dengan css/js *custom* perhalaman
   1. Hanya untuk tahap *development*
   2. Pada saat *production*, file digabung jadi 1 file, menggunakan *tools* (otomatis)
4. Buat folder baru pada view berdasarkan nama controllernya
5. Email template menggunakan CMS yang sudah ada
6. Gunakan *views* & *templating* di dalam applikasi (codeigniter) dan frontend (jquery) menggunakan underscore.js
7. Selalu mengutamakan pengunakan fungsi bawaan dari *codeigniter*

Link: <https://www.codeigniter.com/userguide3/general/welcome.html>

1. Selalu menggunakan POST untuk *request*, hindari menggunakan GET
2. Memperhatikan *request* *time* fungsi yang di gunakan, maksimal. 1 detik
3. Pengelompokan fungsi yang dibuat dan pisahkan antara fungsi yang satu dengan yang lainnya
4. Gunakan *single quote* ($data[‘username’]) karena 7x lebih cepat daripada menggunakan *double quote* ($data[“username”])
5. Selalu gunakan *limit* atau *paging* saat request ke database
6. Selalu membuat *code* sebagai *modular* dan dapat digunakan kembali
7. Jika memungkinkan, gunakan *case statement (switch),* karena 5x lebih cepat daripada menggunakan *if/else*
8. Selalu gunakan *array* multi dimensi daripada menggunakan banyak *variable array*
9. Selalu menggunakan *short code* dan terkadang lebih baik jangan gunakan variable jika parameternya sedikit

Contoh:

**Penggunaan if/else:**

If($data==1){

$value = “ada”;

} else {

$value = “tidak ada”;

}

**Diubah menjadi:**

$value = ($data==1) ? “ada” : “tidak ada”;

**Penggunaan variable:**

items = array(0,1,2,3,4,5);

itemcount = count(items);

print "there are " + itemcount + " items;

**Diubah menjadi:**

items = array(0,1,2,3,4,5);

print "there are " + count(items) + " items";

1. jika menggunakan *regular expressions*, maka gunakan ereg\_replace() dan preg\_replace(), karena akan lebih cepat daripada menggunakan str\_replace().
2. Gunakan isset() daripada menggunakan strlen() untuk mengecek data
3. Batasi jumlah baris dalam 1 line sebanyak 80 karakter
4. Gunakan fungsi yang sudah tersedia dari PHP native

Link: <http://php.net/manual/en/indexes.functions.php>

1. Kurangi pemanggilan data dari database, selalu gunakan kembali data yang sudah dikirim ke database dan di olah kembali
2. Selalu gunakan json untuk manipulasi data
3. *Concatenating variable* lebih cepat daripada hanya menempatkan mereka dalam dua kutip.

Contoh:

$var = 'Hello ' . $world;

1. Selalu gunakan join table daripada membuat banyak request ke database
2. Matikan *debugging notification* sebelum publikasi ke *live server*
3. Jangan menggunakan \* (all) pada query database karena akan semakin lama querynya

# Manajemen Function dan Library

1. Jika menggunakan API third party, taro di library jangan taro di function, misalnya:
   1. Menggunakan API untuk koneksi dengan Mailchimp
   2. Koneksi ke API Doku
2. Jika ada fungsi yang berulang, taro di helper, misalnya:
   1. Fungsi convert date
   2. Currency IDR / number seperator
3. Buat fungsi yang dinamis jika ada kebutuhan yang spesifik (jangan buat fungsi baru), tapi buat default, misalnya:
   1. Format awal date, ddmmyy
   2. Ada kebutuhan jadi, mmddyy
4. Gunakan file constant.php untuk taro nilai-nilai default yang general

# Kerapihan coding

1. Gunakan nama yang jelas dan rapih (tidak ambigu dan memiliki banyak makna) untuk:
   1. Variable
   2. Fungsi
   3. Class
   4. File name
2. Satu baris code panjang maksimal 80 character (disetting dieditor nya, sublime)
3. 1 Tab = 4 spasi
4. Pengelompokan fungsi - fungsi coding dengan comment, dengan format sbb:

/\*\*

\* fungsi untuk membuat generate password

\* \* @author Agung Trilaksono Suwarto Putra <agungtrilaksonosp@gmail.com>

\* @param $password password

\* @param $panjang untuk menentukan berapa panjang karakter dari password

\* @param $character karakter yang di random

\* @param @where (optional) where query tabel

\* @param $terpilih (optional) list terpilih (selected)

\* @return string untuk password siswa dan orang tua

\*

\*/

# Rules

Pilihan Coding Standard

1. Jangan temukan/gunakan standar sendiri.
2. Gunakan standar yang telah ada
3. Gunakan standar yang telah disetujui oleh banyak pihak. Atau pilih standar yang kompatibel dengan library/modul yang akan digunakan
4. Atau gunakan standar sesuai dengan permintaan saat outsourcing (sesuaikan dengan kondisi yang ada)
5. Tetap gunakan satu standar yang dipilih, jangan dicampur antara standar yang satu dengan yang lain

# Referensi Coding Standard

1. PEAR-like coding standard
2. PSR (direkomendasikan)
   1. http://www.php-fig.org/psr/0
   2. http://www.php-fig.org/psr/1
   3. http://www.php-fig.org/psr/2
   4. http://www.php-fig.org/psr/3

# DATABASE

1. Tipe tabel harus InnoDB\* (MySQLi)
2. Setiap tabel harus mempunyai kolom :
   * create\_date (default current timestamp)
   * modify\_date (datetime)
   * user\_id\_create (user
   * user\_id\_modify
   * is\_delete
3. Setiap tabel yang mempunyai relasi harus mempunyai foreign key.
4. foreign key method : restricted
5. penulisan foreign key didahului dengan **id\_[nama\_tabel]**
6. penamaan table meggunakan awalan **ref\_[nama\_table referensi]**
7. penamaan view menggunakan awalan **view\_[nama\_view]**
8. field yang berhubungan dengan tanggal, gunakan date / datetime
9. varchar : *length-*nya disesuaikan dengan kebutuhan data (jangan 255 semua)
10. Normalisasi. struktur database dibuat dengan aturan normalisasi.
11. collation : utf8\_bin
    * agar masuk ke db nya textnya apa adanya.
    * kelemahan : search jadi case sensitive. solusi pake fungsi LCASE(nama\_field)
12. penamaan field menggunakan bahasa inggris
13. query tidak boleh select \*
14. indexing : buat indexing utk field yang tingkat pencariannya tinggi.

f