LAMPIRAN

Lampiran 1 Source Code Alternatif

```
<?php
defined('BASEPATH') or exit('No direct script access allowed');

class Alternatif extends MY_Controller
{
	private $class = 'alternatif';

	public function index()
	{
		$data['alternatif_data'] = $this->getAlternatif();

		$data['title'] = 'Alternatif';
		$this->load->view('templates/header', array('page'
=> $this->class));
		$this->load->view('admin/alternatif/index', $data);
		$this->load->view('templates/footer');
	}

	public function getSubkriteria(){
		return $this->db->get('subkriteria')->result();
	}

	public function getAlternatif(){
		return $this->db->get('alternatif')->result();
	}

	public function getNilaiSubkriteria($id_alternatif) {
		$this->db-
>select('sa.id_alternatif,sa.id_subkriteria,sa.nilai');
		$this->db->join('subkriteria_alternatif sa',
'sa.id_subkriteria=s.id_subkriteria','left');
		$this->db->where('sa.id_alternatif',$id_alternatif);
		$result = $this->db->get('subkriteria s')->result();
		return $result;
	}

	public function insert()
	{
		$this->load->library('form_validation');

		$this->form_validation->set_rules('nama_alternatif',
'nama_alternatif', 'trim|required');
		$this->form_validation->set_rules('nickname',
'nickname', 'trim|required');
		$this->form_validation->set_rules('role', 'role',
'trim|required');
		$this->form_validation->set_rules('alias', 'alias',
'trim|required');
```

```
            if ($this->form_validation->run() == false) {
                  $data['subkriteria'] = $this-
>getSubkriteria();
                  $data['title'] = 'Tambah Alternatif';
                  $this->load->view('templates/header',
array('page' => $this->class));
                  $this->load->view('admin/alternatif/insert',
$data);
                  $this->load->view('templates/footer');
            } else {
                  $set_alternatif = [
                        'nama_alternatif' => $this->input-
>post('nama_alternatif'),
                        'nickname' => $this->input-
>post('nickname'),
                        'role' => $this->input->post('role'),
                        'alias' => $this->input->post('alias'),
                  ];
                  $this->db->insert('alternatif',
$set_alternatif);
                  $id_alternatif = $this->db->insert_id();
                  $this-
>inputSubkriteriaAlternatif($id_alternatif, $this->input-
>post('nilai'));

                  redirect($this->class);
            }
      }

      public function inputSubkriteriaAlternatif($id_alternatif,
$array_nilai){
            $data = array();
            foreach ($array_nilai as $id_subkriteria => $nilai)
{
                  array_push($data, array(
                        'id_subkriteria' => $id_subkriteria,
                        'id_alternatif' => $id_alternatif,
                        'nilai' => $nilai,
                  ));
            }
            $this->db->insert_batch('subkriteria_alternatif',
 $data);
      }

      public function
clearSubkriteriaAlternatif($id_alternatif){
            $this->db->where('id_alternatif', $id_alternatif);
            $this->db->delete('subkriteria_alternatif');
      }

      public function update($id_alternatif)
      {
            $this->load->library('form_validation');
```

```
            $this->form_validation->set_rules('nama_alternatif',
'nama_alternatif', 'trim|required');
            $this->form_validation->set_rules('nickname',
'nickname', 'trim|required');
            $this->form_validation->set_rules('role', 'role',
'trim|required');
            $this->form_validation->set_rules('alias', 'alias',
'trim|required');

            if ($this->form_validation->run() == false) {
                  $alternatif_data = $this->db
                  ->where('id_alternatif', $id_alternatif)
                  ->get('alternatif')
                  ->row(0);

                  $data['alternatif_data'] = $alternatif_data;
                  $subkriteria = $this->getSubkriteria();
                  $nilai_subkriteria = $this-
>getNilaiSubkriteria($id_alternatif);
                  foreach ($subkriteria as $sub_key => $sub) {
                        if
(!isset($nilai_subkriteria[$sub_key])) {
                              array_push($nilai_subkriteria,
(object)array(
                                    'id_alternatif' =>
$id_alternatif,
                                    'id_subkriteria' => $sub-
>id_subkriteria,
                                    'nilai' => 1,
                              ));
                        }
                  }
                  $data['subkriteria'] = $subkriteria;
                  $data['nilai_subkriteria'] =
$nilai_subkriteria;
                  $data['title'] = 'Update Alternatif';
                  $this->load->view('templates/header',
array('page' => $this->class));
                  $this->load->view('admin/alternatif/update',
$data);
                  $this->load->view('templates/footer');
            } else {
                  $set_alternatif = [
                        'nama_alternatif' => $this->input-
>post('nama_alternatif'),
                        'nickname' => $this->input-
>post('nickname'),
                        'role' => $this->input->post('role'),
                        'alias' => $this->input->post('alias'),
                  ];
                  $this->db->where('id_alternatif',
$id_alternatif)->update('alternatif', $set_alternatif);
```

```
                  $this-
>clearSubkriteriaAlternatif($id_alternatif);
                  $this-
>inputSubkriteriaAlternatif($id_alternatif, $this->input-
>post('nilai'));

                  redirect($this->class);
            }
      }

      public function delete($id_alternatif)
      {
            $this->db
            ->where('id_alternatif', $id_alternatif)
            ->delete('alternatif');

            redirect($this->class);
      }
}
```

Lampiran 2 Source Code Kriteria

```
<?php
defined('BASEPATH') or exit('No direct script access allowed');

class Kriteria extends MY_Controller
{
      private $class = 'kriteria';

      public function index()
      {
            $data['title'] = 'Kriteria';
            $data['kriteria_data'] = $this->db->get('kriteria')-
>result();

            $this->load->view('templates/header', array('page' =>
$this->class));
            $this->load->view('admin/kriteria/index', $data);
            $this->load->view('templates/footer');
      }

      public function insert()
      {
            $this->load->library('form_validation');

            $this->form_validation->set_rules('kriteria',
'kriteria', 'trim|required');
            $this->form_validation->set_rules('keterangan',
'keterangan', 'trim|required');


            if ($this->form_validation->run() == false) {
```

```
                  $data['title'] = 'Tambah Kriteria';
                  $this->load->view('templates/header',
array('page' => $this->class));
                  $this->load->view('admin/kriteria/insert',
$data);
                  $this->load->view('templates/footer');
            } else {
                  $set_kriteria = [
                        'kriteria'        =>        $this->input-
>post('kriteria'),
                        'keterangan'       =>       $this->input-
>post('keterangan'),
                  ];
                  $this->db->insert('kriteria', $set_kriteria);

                  redirect($this->class);
            }
      }

      public function update($id_kriteria)
      {
            $this->load->library('form_validation');

            $this->form_validation->set_rules('kriteria',
'kriteria', 'trim|required');
            $this->form_validation->set_rules('keterangan',
'keterangan', 'trim|required');

            if ($this->form_validation->run() == false) {
                  $data['title'] = 'Update Kriteria';
                  $kriteria_data = $this->db
                  ->where('id_kriteria', $id_kriteria)
                  ->get('kriteria')
                  ->row(0);
                  $data['kriteria_data'] = $kriteria_data;
                  $this->load->view('templates/header',
array('page' => $this->class));
                  $this->load->view('admin/kriteria/update',
$data);
                  $this->load->view('templates/footer');
            } else {
                  $set_kriteria = [
                        'kriteria'        =>        $this->input-
>post('kriteria'),
                        'keterangan'       =>       $this->input-
>post('keterangan'),
                  ];
                  $this->db
                  ->where('id_kriteria', $id_kriteria)
                  ->update('kriteria', $set_kriteria);

                  redirect($this->class);
            }
      }
```

```
	public function delete($id_kriteria)
	{
		$this->db
		->where('id_kriteria', $id_kriteria)
		->delete('kriteria');

		redirect($this->class);
	}
}
```

Lampiran 3 Source Code Subkriteria

```
<?php
defined('BASEPATH') or exit('No direct script access allowed');

class Subkriteria extends MY_Controller
{
	private $class = 'subkriteria';

	public function __construct()
	{
		parent::__construct();
		$this->load->model('Subkriteria_model');
	}

	public function index()
	{
		$data = [
			'list' => $this->Subkriteria_model->list(),
			'kriteria'=>		$this->Subkriteria_model-
>get_kriteria(),
			'count_kriteria'=>	$this->Subkriteria_model-
>count_kriteria(),
			'subkriteria'	=>	$this->Subkriteria_model-
>list()
		];
		$data['title'] = 'Subkriteria';
		$this->load->view('templates/header', array('page' =>
$this->class));
		$this->load->view('admin/subkriteria/index', $data);
		$this->load->view('templates/footer');
	}

	public function save()
	{
		$data = [
			'id_kriteria'		=>		$this->input-
>post('id_kriteria'),
			'keterangan'		=>		$this->input-
>post('keterangan'),
			'subkriteria'		=>		$this->input-
>post('subkriteria'),
			'nilai' => $this->input->post('nilai'),
		];
```

```
            $this->form_validation->set_rules('keterangan',
'Keterangan', 'required');
            $this->form_validation->set_rules('subkriteria',
'Subkriteria', 'required');
            $this->form_validation->set_rules('nilai',   'Nilai',
'required');

            if ($this->form_validation->run() != false) {
                  $result       =       $this->Subkriteria_model-
>insert($data);
                  if (!$result) {
                        $this->session->set_flashdata('message',
'Input error');
                  }
            } else {
                  $this->session->set_flashdata('message', 'Semua
Data Harus Diisi');
            }
            redirect($this->class);

      }

      public function update($id_subkriteria)
      {
            $id_subkriteria            =            $this->input-
>post('id_subkriteria');
            $data = array(
                  'id_kriteria'          =>          $this->input-
>post('id_kriteria'),
                  'keterangan'           =>          $this->input-
>post('keterangan'),
                  'subkriteria'          =>          $this->input-
>post('subkriteria'),
                  'nilai' => $this->input->post('nilai')
            );

            $this->Subkriteria_model->update($id_subkriteria,
$data);
            redirect($this->class);
      }

      public function delete($id_subkriteria)
      {
            $this->Subkriteria_model->delete($id_subkriteria);
            redirect($this->class);
      }

}
```

Lampiran 4 Source Code Perhitungan

```
<?php
defined('BASEPATH') or exit('No direct script access allowed');
```

```
class Perhitungan extends MY_Controller
{
	private $class = 'perhitungan';

	public function index($page='bobot_kriteria')
	{
		// kriteria
		$kriteria = $this->getKriteria();
		$subkriteria = $this->getSubkriteria();
		$matriks_kriteria   =   $this->buatMatriks($kriteria,
1,'kriteria_berpasangan');
		$jumlah_matriks_kriteria_h          =          $this-
>hitungJumlahMatriks($matriks_kriteria);
		$jumlah_matriks_kriteria_v          =          $this-
>hitungJumlahMatriks($this->transpose($matriks_kriteria));
		$eigen_vector_kriteria            =            $this-
>hitungEigenVector($matriks_kriteria,
$jumlah_matriks_kriteria_v);
		$lambda_maks                  =                $this-
>lambdaMaks($jumlah_matriks_kriteria_v, $eigen_vector_kriteria);
		$ci = round(($lambda_maks-count($kriteria))/2,5);
		$cr = round($ci/0.58,5);

		// altiernatif
		$alternatif = $this->getAlternatif('support');

		if (isset($_GET['role'])) {
			$role = $_GET['role'];
			if                                (in_array($role,
array('support','rusher','flanker','observer'))) {
				$alternatif           =           $this-
>getAlternatif($role);
				$data['role'] = $role;
			} else {
				$data['role'] = 'support';
			}
		} else {
			$data['role'] = 'support';
		}

		// passing data ke view
		$data['kriteria'] = $kriteria;
		$data['subkriteria'] = $subkriteria;
		$data['matriks_kriteria'] = $matriks_kriteria;
		$data['jumlah_matriks_kriteria_h']                  =
$jumlah_matriks_kriteria_h;
		$data['jumlah_matriks_kriteria_v']                  =
$jumlah_matriks_kriteria_v;
		$data['eigen_vector_kriteria']                      =
$eigen_vector_kriteria;
		$data['lambda_maks'] = $lambda_maks;
		$data['ci'] = $ci;
		$data['cr'] = $cr;
```

```
            // bobot kriteria
            foreach ($kriteria as $key_k => $k) {
                  $matriks_kriteria           =           $this-
>buatMatriks($kriteria, 1,'bobot_kriteria',$k->id_kriteria);
                  $jumlah_matriks_kriteria_h       =       $this-
>hitungJumlahMatriks($matriks_kriteria);
                  $jumlah_matriks_kriteria_v       =       $this-
>hitungJumlahMatriks($this->transpose($matriks_kriteria));
                  $eigen_vector_kriterias        =        $this-
>hitungEigenVector($matriks_kriteria,
$jumlah_matriks_kriteria_v);
                   $lambda_maks              =              $this-
>lambdaMaks($jumlah_matriks_kriteria_v, $eigen_vector_kriterias);
                  $ci                       =round(($lambda_maks-
count($kriteria))/2,5);
                  $cr = round($ci/0.58,5);
                  $data['kriteria_terhadap_kriteria'][$key_k]   =
array(
                        'keterangan' => $k->keterangan,
                        'matriks_kriteria' => $matriks_kriteria,
                        'jumlah_matriks_kriteria_h'            =>
$jumlah_matriks_kriteria_h,
                        'jumlah_matriks_kriteria_v'            =>
$jumlah_matriks_kriteria_v,
                        'eigen_vector_kriteria'                =>
$eigen_vector_kriterias,
                        'lambda_maks' => $lambda_maks,
                        'ci' => $ci,
                        'cr' => $cr,
                  );
            }

            // bobot alternatif
            $supermatriks =  array();
            foreach ($subkriteria as $key_s => $s) {
                  $matriks_alternatif          =          $this-
 >buatMatriks($alternatif,               1,'bobot_alternatif',$s-
>id_subkriteria);
                  $jumlah_matriks_alternatif_v      =      $this-
>hitungJumlahMatriks($this->transpose($matriks_alternatif));
                  $eigen_vector_alternatif        =        $this-
>hitungEigenVector($matriks_alternatif,
$jumlah_matriks_alternatif_v);

      $data['subkriteria_terhadap_alternatif'][$key_s] = array(
                        'keterangan' => $s->keterangan,
                        'matriks_alternatif'                   =>
$matriks_alternatif,
                        'jumlah_matriks_alternatif_v'          =>
$jumlah_matriks_alternatif_v,
                        'eigen_vector_alternatif'              =>
$eigen_vector_alternatif
                  );
```

```
                  $index_kriteria            =            $this-
>findKriteriaIndexById($kriteria, $s->id_kriteria);
                  foreach ($eigen_vector_alternatif as $key_e  =>
$e) {
                        $weighted_supermatriks[$key_e] = round($e
* $eigen_vector_kriteria[$index_kriteria],3);
                  }
                  array_push($supermatriks, array(
                        'unweighted_supermatriks'              =>
$eigen_vector_alternatif,
                        'weighted_supermatriks'                =>
$weighted_supermatriks,
                  ));
            }

            foreach ($this->transpose(array_column($supermatriks,
'weighted_supermatriks')) as $key_limited => $value) {
                  $limited_supermatriks[$key_limited]           =
round((array_sum($value)/count($value))/count($kriteria),3);
            }
            $ordered_values = $limited_supermatriks;
            rsort($ordered_values);
            $ranking = array();
            foreach   ($limited_supermatriks   as   $key_limit   =>
$value) {
                  foreach  ($ordered_values  as  $ordered_key  =>
$ordered_value) {
                        if ($value === $ordered_value) {
                              $key_limit = $ordered_key;
                              break;
                        }
                  }
                  array_push($ranking, array(
                        'rank' => ((int) $key_limit + 1),
                        'alternatif' =>  $alternatif[$key_limit]-
>nickname,
                        'value' =>  $value,
                  ));
            }

            $data['alternatif'] = $alternatif;
            $data['supermatriks'] = $supermatriks;
            $data['limited_supermatriks']                         =
$limited_supermatriks;
            $data['ranking'] = $ranking;


            if ($page=='bobot_kriteria') {

                  $data['title'] = 'Kriteria terhadap Kriteria';
                  $this->load->view('templates/header',
array('page' => 'bobot_kriteria'));
                  $this->load-
>view('admin/perhitungan/bobot_kriteria',$data);
```

```
            } elseif ($page=='bobot_alternatif') {

                  $data['title']    =    'Subkriteria    terhadap
Alternatif';
                  $this->load->view('templates/header',
array('page' => 'bobot_alternatif'));
                  $this->load-
>view('admin/perhitungan/bobot_alternatif',$data);

            } elseif ($page=='supermatriks') {

                  $data['title'] = 'Supermatriks';
                  $this->load->view('templates/header',
array('page' => 'supermatriks'));
                  $this->load-
>view('admin/perhitungan/supermatriks',$data);

            }

            $this->load->view('templates/footer');
      }

      public function getRoles(){
            return $this->db->get('kriteria')->result();
      }

      public function getKriteria(){
            return $this->db->get('kriteria')->result();
      }

      public function getSubkriteria(){
            return $this->db->get('subkriteria')->result();
      }

      public function getAlternatif($role=null){
            if ($role) {
                  $this->db->where('role', $role);
            }
            return $this->db->get('alternatif')->result();
      }

      public     function     buatMatriks($array,     $diagonal,
$perbandingan=false, $id_terhadap=false){
            $pilihan = array(0.5,1,1.5,2,3);
            $matriks_kriteria = array();
            foreach ($array as $key1 => $value1) {
                  foreach ($array as $key2 => $value2) {
                        if ($key1 == $key2){
                              $matriks_kriteria[$key1][$key2]   =
$diagonal;
                        }else{
                              if ($key1<$key2){

      $matriks_kriteria[$key1][$key2]                           =
$pilihan[array_rand($pilihan)];
```

```
                                    if
($perbandingan=='kriteria_berpasangan') {

      $matriks_kriteria[$key1][$key2]          =          $this-
>getNilaiBobotKriteria($value1->id_kriteria,$value2-
>id_kriteria);
                                    }                      elseif
($perbandingan=='bobot kriteria') {

      $matriks kriteria[$key1][$key2]          =          $this-
>getNilaiBobotKriteria($value1->id_kriteria,$value2-
>id_kriteria,$id_terhadap);
                                    }                      elseif
($perbandingan=='bobot_alternatif') {
                                          //
$matriks_kriteria[$key1][$key2]             =             $this-
>getNilaiBobotAlternatif($value1->id_alternatif,$value2-
>id_alternatif,$id_terhadap);

      $matriks kriteria[$key1][$key2]          =          $this-
>getNilaiBobotAlternatif2($value1->id_alternatif,$value2-
>id_alternatif,$id_terhadap);
                                    }

                                    // START sesuai excel  (boleh
dikomen)
                                    // if ($key1==0 &&  $key2==2)
{
                                    //
      $matriks_kriteria[$key1][$key2] = 3;
                                    //  }  elseif  ($key1==1  &&
$key2==2) {
                                    //
      $matriks_kriteria[$key1][$key2] = 3;
                                    // }
                                    //  END  sesuai  excel  (boleh
dikomen)

                              }else{

      $matriks_kriteria[$key1][$key2]                          =
round(1/$matriks_kriteria[$key2][$key1],3);
                              }
                        }
                  }
            }
            return  $matriks_kriteria;
      }

      public function hitungJumlahMatriks($matriks){
            foreach ($matriks as $key1 => $mat) {
                  $jumlah = 0;
                  foreach ($mat as $key2 => $m) {
                        $jumlah += $m;
                  }
```

```
                  $jumlah_matriks[$key1] = round($jumlah,3);
            }
            return $jumlah_matriks;
      }

      public function transpose($array_one) {
            $array_two = [];
            foreach ($array_one as $key => $item) {
                  foreach ($item as $subkey => $subitem) {
                        $array_two[$subkey][$key] = $subitem;
                  }
            }
            return $array_two;
      }

      public         function         hitungEigenVector($matriks,
$jumlah_matriks){
            foreach ($matriks as $key1 => $mat) {
                  $row = 0;
                  foreach ($mat as $key2 => $m) {
                        $row += $m/$jumlah_matriks[$key2];
                  }
                  $eigen[$key1] = round($row/count($matriks), 3);
            }
            return $eigen;
      }

      public function lambdaMaks($jumlah_matriks, $eigen_vector){
            $lambdaMaks = 0;
            foreach ($jumlah_matriks as $key => $jumlah) {
                  $lambdaMaks += $jumlah*$eigen_vector[$key];
            }
            return round($lambdaMaks,5);
      }

      public      function      findKriteriaIndexById($kriterias,
$id_kriteria){
            $index = null;
            foreach($kriterias as $key => $kriteria) {
                  if ($kriteria->id_kriteria == $id_kriteria) {
                        $index = $key;
                        break;
                  }
            }
            return $index;
      }

      public function getNilaiBobotKriteria($id_krit1, $id_krit2,
$id_terhadap=null){
            $this->db->where('id_kriteria1', $id_krit1);
            $this->db->where('id_kriteria2', $id_krit2);
            if ($id_terhadap!=null){
                  $this->db->where('id_terhadap_kriteria',
$id_terhadap);
            }else{
```

```
                  $this->db->where('id_terhadap_kriteria       IS
NULL', null, false);
            }
            $result = $this->db->get('bobot_kriteria')->result();
            if (count($result)>0) {
                  return $result[0]->nilai;
            } else {
                  $pilihan                                        =
array(0.5,0.333,0.25,0.2,0.167,1,2,3,4,5,6,7,8,9);
                  $nilai_random = $pilihan[array_rand($pilihan)];
                  if ($id_terhadap!=null){
                        $this-
>insertNilaiBobotKriteria($id_krit1,                  $id_krit2,
$nilai_random,$id_terhadap);
                  }else{
                        $this-
>insertNilaiBobotKriteria($id_krit1, $id_krit2, $nilai_random);
                  }
                  return $nilai_random;
            }
      }

      public     function     insertNilaiBobotKriteria($id_krit1,
$id_krit2, $nilai, $id_terhadap=null){
            $this->db->insert('bobot_kriteria', array(
                  'id_kriteria1' => $id_krit1,
                  'id_kriteria2' => $id_krit2,
                  'id_terhadap_kriteria' => $id_terhadap,
                  'nilai' => $nilai,
            ));
      }

      public                                                  function
update_bobot_kriteria($id_terhadap_kriteria=null){
            $input = $this->input->post();
            $data = array();
            $this->clearBobotKriteria($id_terhadap_kriteria);
            foreach ($input as $key => $value) {
                  $ids = explode('_', $key);

                  $id_kriteria1 = $ids[1];
                  $id_kriteria2 = $ids[2];

                  $value = array(
                        'id_kriteria1' => $id_kriteria1,
                        'id_kriteria2' => $id_kriteria2,
                        'nilai' => $value,
                  );
                  if ($id_terhadap_kriteria) {
                        $value['id_terhadap_kriteria']          =
$id_terhadap_kriteria;
                  }
                  array_push($data, $value);
            }
            $this->db->insert_batch('bobot_kriteria', $data);
```

```
            redirect('perhitungan/index/bobot_kriteria');
      }

      public                                                  function
clearBobotKriteria($id_terhadap_kriteria=null){
            if ($id_terhadap_kriteria) {
                  $this->db->where('id_terhadap_kriteria',
$id_terhadap_kriteria);
            } else {
                  $this->db->where('id_terhadap_kriteria       IS
NULL', null, false);
            }
            $this->db->delete('bobot_kriteria');
      }



      public     function     getNilaiBobotAlternatif($id_alter1,
$id_alter2, $id_terhadap=null){
            $this->db->where('id_alternatif1', $id_alter1);
            $this->db->where('id_alternatif2', $id_alter2);
            if ($id_terhadap!=null){
                  $this->db->where('id_subkriteria',
$id_terhadap);
            }
            $result     =     $this->db->get('bobot_alternatif')-
>result();
            if (count($result)>0) {
                  return $result[0]->nilai;
            } else {
                  $pilihan                                        =
array(0.5,0.333,0.25,0.2,0.167,1,2,3,4,5,6,7,8,9);
                  $nilai_random = $pilihan[array_rand($pilihan)];
                  if ($id_terhadap!=null){
                        $this-
>insertNilaiBobotAlternatif($id_alter1,               $id_alter2,
$nilai_random,$id_terhadap);
                  }else{
                        $this-
>insertNilaiBobotAlternatif($id_alter1,               $id_alter2,
$nilai_random);
                  }
                  return $nilai_random;
            }
      }

      public    function    getNilaiBobotAlternatif2($id_alter1,
$id_alter2, $id_terhadap=null){
            $nilai_alter1                =                $this-
>getSubkriteriaAlternatif($id_terhadap, $id_alter1);
            $nilai_alter2                 =               $this-
>getSubkriteriaAlternatif($id_terhadap, $id_alter2);
            return                                        $this-
>getSkalaKepentingan($nilai_alter1,$nilai_alter2);
```

```
	}

	public function getSubkriteriaAlternatif($id_subkriteria, $id_alternatif){
		$this->db->where('id_subkriteria', $id_subkriteria);
		$this->db->where('id_alternatif', $id_alternatif);
		$result = $this->db->get('subkriteria_alternatif')->result();
		if (count($result)>0) {
			return $result[0]->nilai;
		}else{
			return 3;
		}
	}

	public function getSkalaKepentingan($nilai1, $nilai2){
		if ($nilai1==5 && $nilai2==5) {
			return 1;
		} elseif ($nilai1==5 && $nilai2==4) {
			return 2;
		} elseif ($nilai1==5 && $nilai2==3) {
			return 3;
		} elseif ($nilai1==5 && $nilai2==2) {
			return 5;
		} elseif ($nilai1==5 && $nilai2==1) {
			return 7;
		} elseif ($nilai1==4 && $nilai2==3) {
			return 2;
		} elseif ($nilai1==4 && $nilai2==2) {
			return 3;
		} elseif ($nilai1==4 && $nilai2==1) {
			return 5;
		} elseif ($nilai1==3 && $nilai2==2) {
			return 2;
		} elseif ($nilai1==3 && $nilai2==1) {
			return 4;
		} elseif ($nilai1==2 && $nilai2==1) {
			return 2;
		} else {
			return 1;
		}
	}

	public function insertNilaiBobotAlternatif($id alter1, $id_alter2, $nilai, $id_subkriteria=null){
		$this->db->insert('bobot_alternatif', array(
			'id_alternatif1' => $id_alter1,
			'id_alternatif2' => $id_alter2,
			'id_subkriteria' => $id_subkriteria,
			'nilai' => $nilai,
		));
	}

	public function update_bobot_alternatif($id_subkriteria){
		$input = $this->input->post();
```

```
            $data = array();
            foreach ($input as $key => $value) {
                  $ids = explode('_', $key);

                  $id_alternatif1 = $ids[1];
                  $id_alternatif2 = $ids[2];

                  $this-
>clearBobotAlternatifBySubkriteria($id_alternatif1,$id_alternati
f2,$id_subkriteria);
                  array_push($data, array(
                        'id_alternatif1' => $id_alternatif1,
                        'id_alternatif2' => $id_alternatif2,
                        'id_subkriteria' => $id_subkriteria,
                        'nilai' => $value,
                  ));
            }
            $this->db->insert_batch('bobot_alternatif', $data);

            redirect('perhitungan/index/bobot_alternatif');
      }

      public                                                function
clearBobotAlternatifBySubkriteria($id_alternatif1,
$id alternatif2, $id subkriteria){
            $this->db->where('id_alternatif1', $id_alternatif1);
            $this->db->where('id_alternatif2', $id_alternatif2);
            $this->db->where('id_subkriteria', $id_subkriteria);
            $this->db->delete('bobot_alternatif');
      }
}
```