

BAB IV

HASIL DAN PEMBAHASAN

4.1 Pengembangan Sistem

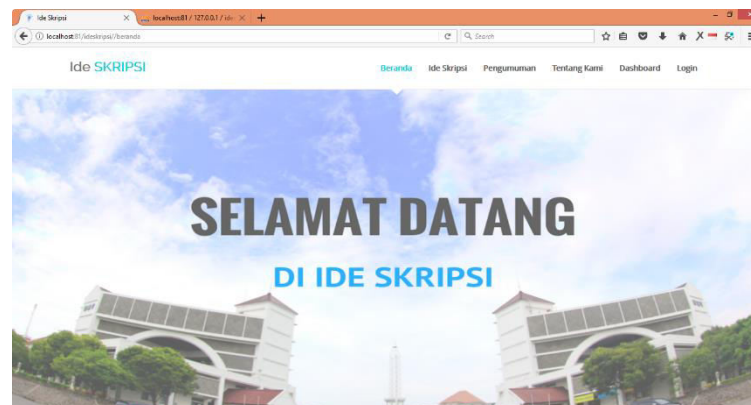
Sistem pengambilan ide skripsi berbasis *website* dikembangkan dengan menggunakan Bahasa pemrograman PHP dan *framework* Codeigniter. Codeigniter menggunakan metode MVC, meliputi *Model*, *View*, dan *Controller*. *Model* berisi fungsi-fungsi untuk mengakses *database*. *View* berfungsi untuk mengatur antarmuka pengguna. *Controller* berfungsi sebagai penghubung antara *model* dan *view*.

4.2 Pembuatan Sistem Pengambilan Ide Skripsi

Berdasarkan rancangan yang telah dibahas pada bab 3, maka dibuatlah sistem pengambilan ide skripsi menggunakan Bahasa pemrograman PHP dan *framework* Codeigniter.

4.2.1 Halaman Beranda

Beranda adalah tampilan awal ketika pengguna mengakses *website*. Tampilan beranda dapat dilihat pada Gambar 4.1.



Gambar 4. 1 Tampilan beranda

```

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

class Beranda extends CI_Controller {

    public function index()
    {
        $this->load->view('beranda');
    }
}

```

Gambar 4. 2 *Coding controller* beranda

```

<?php
$currentPage= 'beranda';
include('/layout/header.php');
?>
<section id="imgberanda">
</section>

<?php include('/layout/footer.php'); ?>

```

Gambar 4. 3 *Coding tampilan* beranda

Gambar 4.2 memperlihatkan *coding controller* Beranda yang memanggil tampilan beranda dalam folder *views*. *Coding* tampilan beranda dapat dilihat pada Gambar 4.3. Pada *coding* tampilan beranda ini, dapat dilihat bahwa tampilan beranda menggunakan *header* dan *footer* yang dipanggil melalui fungsi *include*.

4.2.2 Halaman Registrasi Mahasiswa

Halaman registrasi mahasiswa adalah halaman dimana mahasiswa dapat mendaftarkan akun. Mahasiswa harus mengisi nama, NIM, prodi, fakultas, alamat, nomor hp, jumlah sks, IPK, username, email, dan password. Tampilan halaman registrasi mahasiswa dapat dilihat pada Gambar 4.4.

The image shows a web browser window with a red title bar. The address bar shows 'localhost:81/ide/skrpsi'. The main content area has a blue background. In the center, there is a white registration form titled 'Pendaftaran Akun Mahasiswa'. The form contains the following fields from top to bottom: a text input for 'Nama', a text input for 'NIM', a dropdown menu for 'Prodi', a dropdown menu for 'Fakultas', a text input for 'Alamat', a text input for 'No. HP', a text input for 'Jumlah SKS', and a text input for 'IPK (contoh: 3.5)'. There is a small 'Kembali' button at the bottom left of the form.

Gambar 4. 4 Tampilan registrasi mahasiswa

Coding controller registrasi mahasiswa dapat dilihat pada Gambar 4.5. Dalam fungsi *index* dapat dilihat bahwa sistem melakukan validasi input, yaitu input harus diisi, panjang minimal dan maksimal, input unik, input *integer*, pencocokan input, dan validasi *email*. Jika validasi benar, data akan dimasukkan ke dalam *database* melalui fungsi *daftar* yang ada dalam *M_account* pada folder *views*. *Coding* fungsi *daftar* dapat dilihat pada Gambar 4.6.

Pada Gambar 4.8 dapat dilihat *coding* tampilan register mahasiswa. Pada input NIM, nomor HP, jumlah SKS, dan IPK terdapat atribut *onkeypress* yang mana pengguna hanya dapat mengetikkan karakter yang telah ditentukan dalam javascript pada Gambar 4.7.

```

public function index() {

$this->form_validation->set_rules('name', 'NAME', 'required|max_length[30]');
$this->form_validation->set_rules('nim', 'NIM', 'required|trim|integer|is_unique[mahasiswa.NIM_MHS]|max_length[11]',
    array(
        'is_unique' => 'This %s already exists.'));
$this->form_validation->set_rules('prodi', 'PRODI', 'required');
$this->form_validation->set_rules('fakultas', 'FAKULTAS', 'required');
$this->form_validation->set_rules('alamat', 'ALAMAT', 'required');
$this->form_validation->set_rules('nohp', 'NOHP', 'required|trim|integer|is_unique[mahasiswa.NOHP_MHS]|max_length[12]',
    array(
        'is_unique' => 'This %s already exists.'));
$this->form_validation->set_rules('sks', 'SKS', 'required|integer|max_length[3]');
$this->form_validation->set_rules('ipk', 'IPK', 'required');
$this->form_validation->set_rules('username', 'USERNAME', 'required|trim|min_length[4]|max_length[15]|is_unique[mahasiswa.USERNAME_MH]',
    array(
        'is_unique' => 'This %s already exists.'));
$this->form_validation->set_rules('email', 'EMAIL', 'required|valid_email|is_unique[mahasiswa.EMAIL_MHS]|max_length[30]',
    array(
        'is_unique' => 'This %s already exists.'));
$this->form_validation->set_rules('password', 'PASSWORD', 'required|min_length[5]|max_length[50]');
$this->form_validation->set_rules('password_conf', 'CONFIRM PASSWORD', 'required|matches[password]');
if($this->form_validation->run() == FALSE) {
    $this->load->view('account/v_register');
} else {
    $data['nama_mhs'] = $this->input->post('name');
    $data['nim_mhs'] = $this->input->post('nim');
    $data['prodi_mhs'] = $this->input->post('prodi');
    $data['fakultas_mhs'] = $this->input->post('fakultas');
    $data['alamat_mhs'] = $this->input->post('alamat');
    $data['nohp_mhs'] = $this->input->post('nohp');
    $data['sks_mhs'] = $this->input->post('sks');
    $data['ipk_mhs'] = $this->input->post('ipk');
    $data['username_mhs'] = $this->input->post('username');
    $data['email_mhs'] = $this->input->post('email');
    $data['password_mhs'] = md5($this->input->post('password'));
    $data['id_users'] = $this->input->post('idusers');

    $this->m_account->daftar($data);

    $pesan['message'] = "Pendaftaran berhasil";

    $this->load->view('account/v_success', $pesan);
}
}

```

Gambar 4.5 Coding register mahasiswa

```

function daftar($data)
{
    $this->db->insert('mahasiswa', $data);
}

```

Gambar 4.6 Coding fungsi daftar mahasiswa

```

<script type="text/javascript">
function isNumber(evt) {
    evt = (evt) ? evt : window.event;
    var charCode = (evt.which) ? evt.which : evt.keyCode;
    if (charCode > 31 && (charCode < 48 || charCode > 57)) {
        return false;
    }
    return true;
}
function isFloat(evt) {
    var theEvent = evt || window.event;
    var key = theEvent.keyCode || theEvent.which;
    key = String.fromCharCode(key);
    if (key.length == 0) return;
    var regex = /^[0-9.\b]+$/;
    if (!regex.test(key)) {
        theEvent.returnValue = false;
        if (theEvent.preventDefault) theEvent.preventDefault();
    }
}
}
</script>

```

Gambar 4.7 Coding javascript pada register mahasiswa

```

<section class="container">
  <div class="login">
    <h1>Pendaftaran Akun Mahasiswa</h1>
    <form method="post">

      <?php echo form_open('register');?>

      <p>
        <input type="text" name="name" placeholder="Nama" value="<?php echo set_value('name'); ?>" required />
      </p>
      <p> <?php echo form_error('name'); ?> </p>

      <p>
        <input type="text" name="nim" placeholder="NIM" value="<?php echo set_value('nim'); ?>" onkeypress="return isNumber(event)" required />
      </p>
      <p> <?php echo form_error('nim'); ?> </p>

      <p>
        <select name="prodi" class="form-control" required>
          <option value="" disabled selected style="display: none;">Prodi</option>
          <option value="Teknologi Informasi">Teknologi Informasi</option>
        </select>
      </p>
      <p> <?php echo form_error('prodi'); ?> </p>

      <p>
        <select name="fakultas" class="form-control" required>
          <option value="" disabled selected style="display: none;">Fakultas</option>
          <option value="Teknik">Teknik</option>
        </select>
      </p>
      <p> <?php echo form_error('fakultas'); ?> </p>

      <p>
        <input type="text" name="alamat" placeholder="Alamat" value="<?php echo set_value('alamat'); ?>" required />
      </p>
      <p> <?php echo form_error('alamat'); ?> </p>

      <p>
        <input type="text" name="nohp" placeholder="No. HP" value="<?php echo set_value('nohp'); ?>" onkeypress="return isNumber(event)" requi
      </p>
      <p> <?php echo form_error('nohp'); ?> </p>

      <p>
        <input type="text" name="sks" placeholder="Jumlah SKS" value="<?php echo set_value('sks'); ?>" onkeypress="return isNumber(event)" req
      </p>
      <p> <?php echo form_error('sks'); ?> </p>

      <p>
        <input type="text" name="ipk" placeholder="IPK (contoh: 3.5)" value="<?php echo set_value('ipk'); ?>" onkeypress="return isFloat(event
      </p>
      <p> <?php echo form_error('ipk'); ?> </p>

      <p>
        <input type="text" name="username" placeholder="Username" value="<?php echo set_value('username'); ?>" required />
      </p>
      <p> <?php echo form_error('username'); ?> </p>

      <p>
        <input type="text" name="email" placeholder="Email" value="<?php echo set_value('email'); ?>" required />
      </p>
      <p> <?php echo form_error('email'); ?> </p>

      <p>
        <input type="password" name="password" placeholder="Password" value="<?php echo set_value('password'); ?>" required />
      </p>
      <p> <?php echo form_error('password'); ?> </p>

      <p>
        <input type="password" name="password_conf" placeholder="Confirm Password" value="<?php echo set_value('password_conf'); ?>" required
      </p>
      <p> <?php echo form_error('password_conf'); ?> </p>

      <p>
        <input type="hidden" name="idusers" value="2"/>
      </p>

      <p class="submit">
        <input type="submit" name="btnSubmit" value="Daftar" />
      </p>

```

Gambar 4. 8 Coding tampilan register mahasiswa

4.2.3 Halaman Registrasi Dosen

Halaman registrasi dosen adalah halaman dimana dosen dapat mendaftarkan akun. Dosen harus mengisi nama, NIDN, prodi, fakultas, alamat, nomor hp, pendidikan terakhir, keahlian, username, email, dan password. Tampilan halaman registrasi dosen dapat dilihat pada Gambar 4.9.

Gambar 4. 9 Tampilan registrasi dosen

Coding controller registrasi dosen dapat dilihat pada Gambar 4.10. Dalam fungsi *index* dapat dilihat bahwa sistem melakukan validasi input, yaitu input harus diisi, panjang minimal dan maksimal, input unik, input *integer*, pencocokan input, dan validasi *email*. Jika validasi benar, data akan dimasukkan ke dalam *database* melalui fungsi *daftardosen* yang ada dalam *M_account* pada folder *views*. *Coding* fungsi *daftar* dapat dilihat pada Gambar 4.11.

Pada Gambar 4.13 dapat dilihat *coding* tampilan register mahasiswa. Pada input NIDN dan nomor HP terdapat atribut *onkeypress* yang mana pengguna hanya dapat mengetikkan karakter yang telah ditentukan dalam javascript pada Gambar 4.12.

```

public function index() {
    $this->form_validation->set_rules('nidn', 'NIDN', 'required|trim|integer|is_unique[dosen.NIDN_DSN]|max_length[10]',
        array(
            'is_unique' => 'This %s already exists.'));
    $this->form_validation->set_rules('name', 'NAME', 'required|max_length[30]');
    $this->form_validation->set_rules('prodi', 'PRODI', 'required');
    $this->form_validation->set_rules('fakultas', 'FAKULTAS', 'required');
    $this->form_validation->set_rules('alamat', 'ALAMAT', 'required');
    $this->form_validation->set_rules('nohp', 'NOHP', 'required|is_unique[dosen.NOHP_DSN]|max_length[12]',
        array(
            'is_unique' => 'This %s already exists.'));
    $this->form_validation->set_rules('pendidikanterakhir', 'PENDIDIKANTERAKHIR', 'required');
    $this->form_validation->set_rules('keahlian', 'Keahlian', 'required|max_length[100]');
    $this->form_validation->set_rules('username', 'USERNAME', 'required|trim|min_length[4]|max_length[15]|is_unique[dosen.USERNAME_DSN]',
        array(
            'is_unique' => 'This %s already exists.'));
    $this->form_validation->set_rules('email', 'EMAIL', 'required|valid_email|is_unique[dosen.EMAIL_DSN]|max_length[30]',
        array(
            'is_unique' => 'This %s already exists.'));
    $this->form_validation->set_rules('password', 'PASSWORD', 'required|min_length[5]|max_length[50]');
    $this->form_validation->set_rules('password_conf', 'CONFIRM PASSWORD', 'required|matches[password]');
    if($this->form_validation->run() == FALSE) {
        $this->load->view('account/v_regdosen');
    }else{
        $data['nidn_dsn'] = $this->input->post('nidn');
        $data['nama_dsn'] = $this->input->post('name');
        $data['prodi_dsn'] = $this->input->post('prodi');
        $data['fakultas_dsn'] = $this->input->post('fakultas');
        $data['alamat_dsn'] = $this->input->post('alamat');
        $data['nohp_dsn'] = $this->input->post('nohp');
        $data['pendidikan_dsn'] = $this->input->post('pendidikanterakhir');
        $data['keahlian_dsn'] = $this->input->post('keahlian');
        $data['username_dsn'] = $this->input->post('username');
        $data['email_dsn'] = $this->input->post('email');
        $data['password_dsn'] = md5($this->input->post('password'));
        $data['id_users'] = $this->input->post('id_users');

        $this->m_account->daftardosen($data);

        $pesan['message'] = "Pendaftaran berhasil";

        $this->load->view('account/v_success',$pesan);
    }
}

```

Gambar 4. 10 Coding register dosen

```

function daftardosen($data)
{
    : : $this->db->insert('dosen',$data);
}

```

Gambar 4. 11 Coding fungsi daftar dosen

```

<script type="text/javascript">
function isNumber(evt) {
    evt = (evt) ? evt : window.event;
    var charCode = (evt.which) ? evt.which : evt.keyCode;
    if (charCode > 31 && (charCode < 48 || charCode > 57)) {
        return false;
    }
    return true;
}
</script>

```

Gambar 4. 12 Coding javascript pada register dosen

```

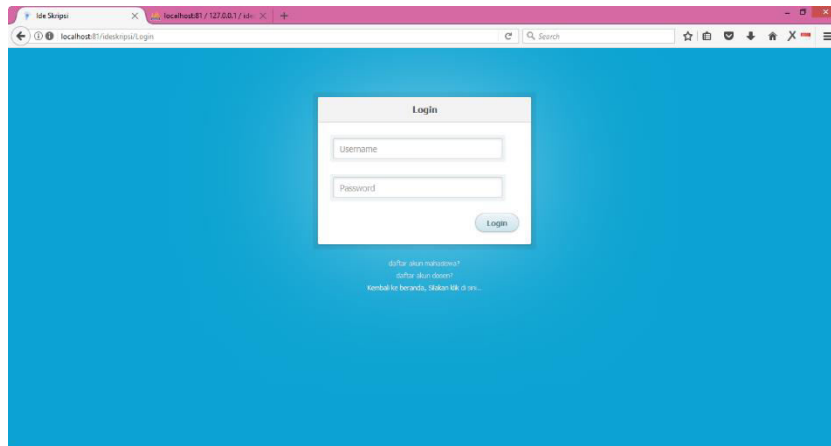
<section class="container">
  <div class="login">
    <h1>Pendaftaran Akun Dosen</h1>
    <form method="post">
      <?php echo form_open('register');?>
      <p>
        <input type="text" name="name" placeholder="Nama" value="<?php echo set_value('name'); ?>" required />
        </p>
        <p> <?php echo form_error('name'); ?> </p>
      <p>
        <input type="text" name="nidn" placeholder="NIDN" value="<?php echo set_value('nidn'); ?>" onkeypress="return isNumber(event)" required />
        </p>
        <p> <?php echo form_error('nidn'); ?> </p>
      <p>
        <select name="prodi" class="form-control" required>
          <option value="" disabled selected style="display: none;">Prodi</option>
          <option value="Teknologi Informasi">Teknologi Informatika</option>
        </select>
        </p>
        <p> <?php echo form_error('prodi'); ?> </p>
      <p>
        <select name="fakultas" class="form-control" required>
          <option value="" disabled selected style="display: none;">Fakultas</option>
          <option value="Teknik">Teknik</option>
        </select>
        </p>
        <p> <?php echo form_error('fakultas'); ?> </p>
      <p>
        <input type="text" name="alamat" placeholder="Alamat" value="<?php echo set_value('alamat'); ?>" required />
        </p>
        <p> <?php echo form_error('alamat'); ?> </p>
      <p>
        <input type="text" name="nohp" placeholder="No. HP" value="<?php echo set_value('nohp'); ?>" onkeypress="return isNumber(event)" required />
        </p>
        <p> <?php echo form_error('nohp'); ?> </p>
      <p>
        <select name="pendidikanterakhir" class="form-control" required>
          <option value="" disabled selected style="display: none;">Pendidikan Terakhir</option>
          <option value="S1">S1</option>
          <option value="S2">S2</option>
          <option value="S3">S3</option>
        </select>
        </p>
        <p> <?php echo form_error('pendidikanterakhir'); ?> </p>
      <p>
        <input type="text" name="keahlian" placeholder="Keahlian" value="<?php echo set_value('keahlian'); ?>" required />
        </p>
        <p> <?php echo form_error('keahlian'); ?> </p>
      <p>
        <input type="text" name="username" placeholder="Username" value="<?php echo set_value('username'); ?>" required />
        </p>
        <p> <?php echo form_error('username'); ?> </p>
      <p>
        <input type="text" name="email" placeholder="Email" value="<?php echo set_value('email'); ?>" required />
        </p>
        <p> <?php echo form_error('email'); ?> </p>
      <p>
        <input type="password" name="password" placeholder="Password" value="<?php echo set_value('password'); ?>" required />
        </p>
        <p> <?php echo form_error('password'); ?> </p>
      <p>
        <input type="password" name="password_conf" placeholder="Confirm Password" value="<?php echo set_value('password_conf'); ?>" required />
        </p>
        <p> <?php echo form_error('password_conf'); ?> </p>
      <p>
        <input type="hidden" name="id_users" value="3"/>
        </p>
    </form>
  </div>
</section>

```

Gambar 4.13 Coding tampilan register dosen

4.2.4 Halaman *Login*

Halaman *login* berfungsi untuk memberikan keamanan di dalam sistem. Pengguna harus mengisi *username* dan *password* yang benar untuk dapat masuk ke dalam sistem. Tampilan halaman *login* dapat dilihat pada Gambar 4.14.



Gambar 4. 14 Tampilan halaman *login*

Pada Gambar 4.15, dapat dilihat bahwa sistem melakukan validasi *login*. Jika *username* dan *password* sudah diisi, sistem akan melakukan validasi *login* menggunakan fungsi *login* dalam *library* *simple_login*. Coding fungsi *login* dapat dilihat pada Gambar 4.16.

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

class Login extends CI_Controller {

    public function index() {

        // Fungsi Login
        $valid = $this->form_validation;
        $username = $this->input->post('username');
        $password = $this->input->post('password');
        $valid->set_rules('username','Username','required');
        $valid->set_rules('password','Password','required');

        if($valid->run()) {
            $this->simple_login->login($username,$password);
        }
        // End fungsi login
        $this->load->view('account/v_login');
    }

    public function logout(){
        $this->simple_login->logout();
    }
}
```

Gambar 4. 15 Coding controller *Login*

```

public function login($username, $password) {
    //cek username dan password
    $query = $this->CI->db->get_where('mahasiswa',array('username_mhs'=>$username,'password_mhs' => md5($password)));
    $query2 = $this->CI->db->get_where('dosen',array('username_dsn'=>$username,'password_dsn' => md5($password)));
    $query3 = $this->CI->db->get_where('admin',array('username_admin'=>$username,'password_admin' => md5($password)));

    if($query->num_rows() == 1) {
        //ambil data user berdasar username
        $row = $this->CI->db->query('SELECT nim_mhs FROM mahasiswa where username_mhs = "'.$username.'"');
        $row2 = $this->CI->db->query('SELECT id_users FROM mahasiswa where username_mhs = "'.$username.'"');
        $row3 = $this->CI->db->query('SELECT nama_mhs FROM mahasiswa where username_mhs = "'.$username.'"');

        $admin = $row->row();
        $tipe = $row2->row();
        $nama = $row3->row();

        $id = $admin->nim_mhs;
        $tipeuser = $tipe->id_users;
        $namauser = $nama->nama_mhs;

        //set session user
        $this->CI->session->set_userdata('username', $username);
        $this->CI->session->set_userdata('id_login', uniqid(rand()));
        $this->CI->session->set_userdata('id_user', $id);
        $this->CI->session->set_userdata('tipeuser', $tipeuser);
        $this->CI->session->set_userdata('namauser', $namauser);

        //redirect ke halaman dashboard
        redirect(site_url('dashboard'));
    }
    elseif($query2->num_rows() == 1) {
        //ambil data user berdasar username
        $row = $this->CI->db->query('SELECT nidn_dsn FROM dosen where username_dsn = "'.$username.'"');
        $row2 = $this->CI->db->query('SELECT id_users FROM dosen where username_dsn = "'.$username.'"');
        $row3 = $this->CI->db->query('SELECT nama_dsn FROM dosen where username_dsn = "'.$username.'"');

        $admin = $row->row();
        $tipe = $row2->row();
        $nama = $row3->row();

        $id = $admin->nidn_dsn;
        $tipeuser = $tipe->id_users;
        $namauser = $nama->nama_dsn;

        //set session user
        $this->CI->session->set_userdata('username', $username);
        $this->CI->session->set_userdata('id_login', uniqid(rand()));
        $this->CI->session->set_userdata('id_user', $id);
        $this->CI->session->set_userdata('tipeuser', $tipeuser);
        $this->CI->session->set_userdata('namauser', $namauser);

        //redirect ke halaman dashboard
        redirect(site_url('dashboard'));
    }
    elseif($query3->num_rows() == 1) {
        //ambil data user berdasar username
        $row = $this->CI->db->query('SELECT id_admin FROM admin where username_admin = "'.$username.'"');
        $row2 = $this->CI->db->query('SELECT id_users FROM admin where username_admin = "'.$username.'"');
        $row3 = $this->CI->db->query('SELECT nama_admin FROM admin where username_admin = "'.$username.'"');

        $admin = $row->row();
        $tipe = $row2->row();
        $nama = $row3->row();

        $id = $admin->id_admin;
        $tipeuser = $tipe->id_users;
        $namauser = $nama->nama_admin;

        //set session user
        $this->CI->session->set_userdata('username', $username);
        $this->CI->session->set_userdata('id_login', uniqid(rand()));
        $this->CI->session->set_userdata('id_user', $id);
        $this->CI->session->set_userdata('tipeuser', $tipeuser);
        $this->CI->session->set_userdata('namauser', $namauser);

        //redirect ke halaman dashboard
        redirect(site_url('dashboard'));
    }
    else{
        //jika tidak ada, set notifikasi dalam flashdata.
        $this->CI->session->set_flashdata('sukses','Username atau password anda salah, silakan coba lagi.. ');

        //redirect ke halaman login
        redirect(site_url('login'));
    }
    return false;
}
}

```

Gambar 4.16 Coding fungsi login

Jika validasi *login* benar, sistem mengalihkan ke *controller dashboard*. *Coding controller dashboard* dapat dilihat pada Gambar 4.17. Pada fungsi *index*, dapat dilihat bahwa tiap tipe akun diarahkan ke halaman *dashboard* masing-masing.

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

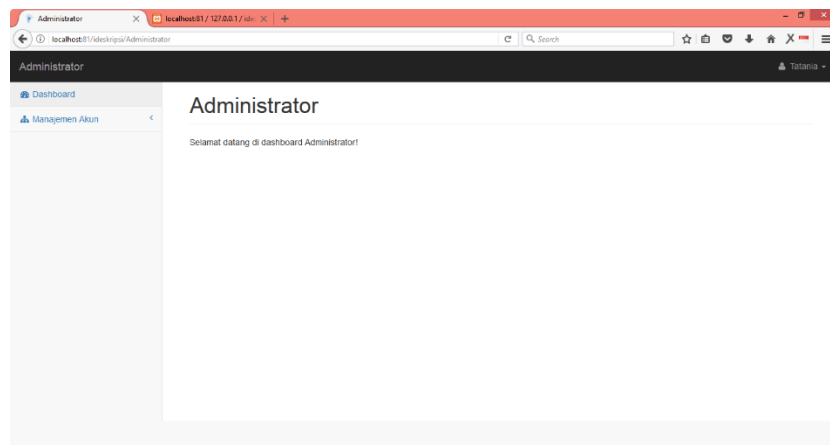
class Dashboard extends CI_Controller{
    function __construct(){
        parent::__construct();
        $this->simple_login->cek_login();
    }
    //load halaman dashboard
    public function index(){

        if($this->session->userdata('tipeuser')== '2') {
            redirect(site_url('Mahasiswa'));
        }
        elseif($this->session->userdata('tipeuser') == '3') {
            redirect(site_url('Dosen'));
        }
        elseif($this->session->userdata('tipeuser') == '1') {
            redirect(site_url('Administrator'));
        }
    }
}
}
```

Gambar 4. 17 Coding controller dashboard

4.2.5 Halaman *Dashboard Administrator*

Halaman *dashboard* administrator adalah halaman awal yang tertampil ketika administrator melakukan *login* ke dalam sistem. Pada *dashboard* administrator, terdapat menu manajemen akun mahasiswa dan dosen. Tampilan halaman *dashboard* administrator dapat dilihat pada Gambar 4.18.

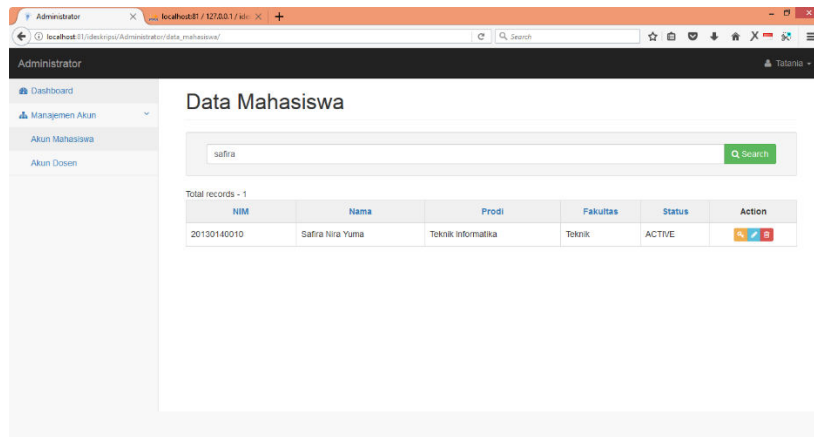


Gambar 4. 18 Tampilan *dashboard* administrator

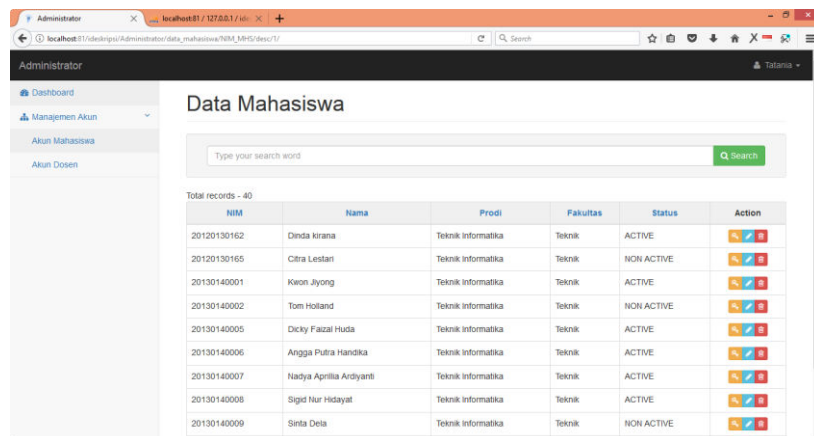
Pada menu manajemen akun mahasiswa, Administrator dapat melihat data akun mahasiswa, melakukan pencarian, pengurutan data, aktivasi akun, mengubah *password*, dan menghapus akun mahasiswa. Tampilan dapat dilihat pada Gambar 4.19, Gambar 4.20, Gambar 4.21, Gambar 4.22, Gambar 4,23, Gambar 4.24.

NIM	Nama	Prodi	Fakultas	Status	Action
20130140006	Angga Putra Handika	Teknik Informatika	Teknik	ACTIVE	Edit Delete
20130140276	Anggita Nur	Teknik Informatika	Teknik	NON ACTIVE	Edit Delete
20130140130	Bachsar Madya Perma	Teknik Informatika	Teknik	ACTIVE	Edit Delete
20120130165	Citra Lestari	Teknik Informatika	Teknik	NON ACTIVE	Edit Delete
20130140005	Dicky Faizal Huda	Teknik Informatika	Teknik	ACTIVE	Edit Delete
20120130162	Dinda Kirana	Teknik Informatika	Teknik	ACTIVE	Edit Delete
20130140139	Dirga Rama Sudira	Teknik Informatika	Teknik	ACTIVE	Edit Delete
20130140134	Fauzi Azhari	Teknik Informatika	Teknik	NON ACTIVE	Edit Delete
20130140013	Fathania Firwan Firdaus	Teknik Informatika	Teknik	ACTIVE	Edit Delete

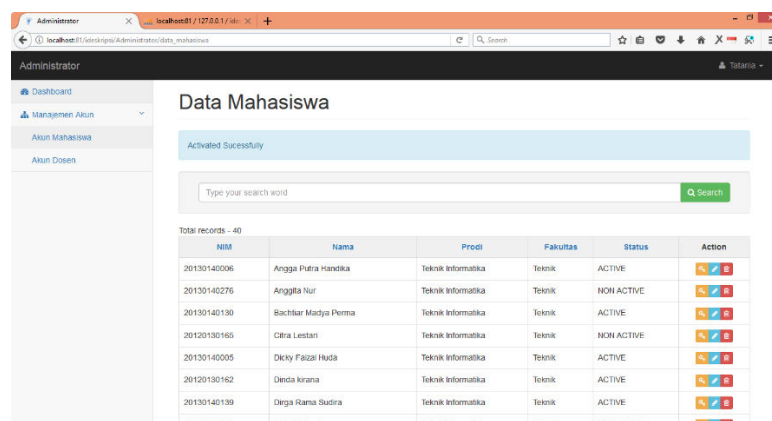
Gambar 4. 19 Tampilan manajemen akun mahasiswa



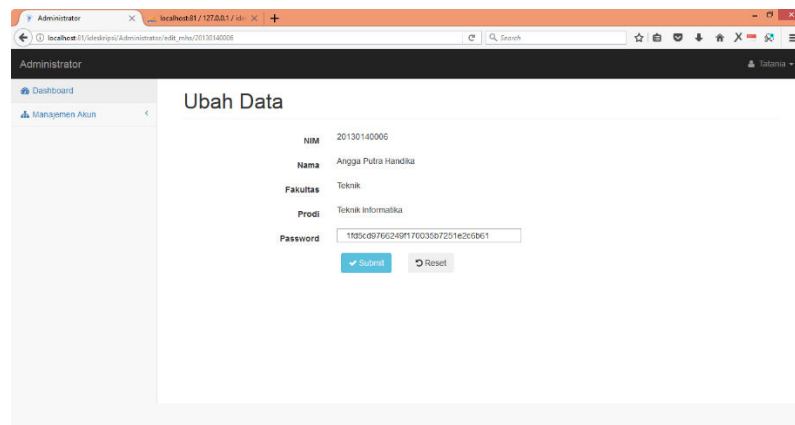
Gambar 4. 20 Tampilan pencarian data mahasiswa



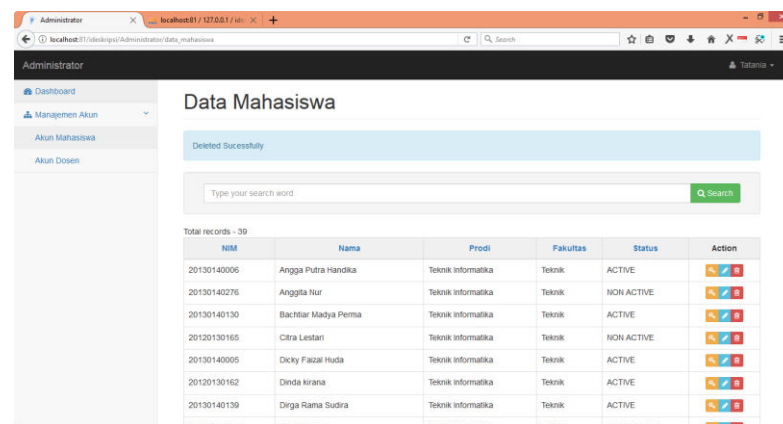
Gambar 4. 21 Tampilan data mahasiswa urut berdasarkan NIM



Gambar 4. 22 Tampilan berhasil aktivasi akun mahasiswa



Gambar 4. 23 Tampilan ubah *password* akun mahasiswa



Gambar 4. 24 Tampilan berhasil menghapus akun mahasiswa

Pada Gambar 4.25, dapat dilihat bahwa fungsi `data_mahasiswa` memanggil fungsi `get_mahasiswa` dan `record_count` yang ada pada `M_Mhs` dalam folder `models`. Fungsi `get_mahasiswa` digunakan untuk mendapatkan data mahasiswa dari `database`. Sedangkan fungsi `record_count` digunakan untuk menghitung jumlah data mahasiswa. *Coding* fungsi `get_mahasiswa` dan `record_count` dapat dilihat pada Gambar 4.26.

```

public function data_mahasiswa()
{
    $data = array();
    $data['sort_cols'] = array(
        'NIM_MHS' => 'NIM',
        'NAMA_MHS' => 'Nama',
        'PRODI_MHS' => 'Prodi',
        'FAKULTAS_MHS' => 'Fakultas',
        'STATUS' => 'Status'
    );
    $config = array();
    //base_url () - 'index.php/questions/page/'.$sortfield.'/'.$order.'/';
    $search_string = $this->input->post('search');

    $config["per_page"] = 10;
    //max number of page links
    $config['num_links'] = 2;
    //use page number as parameter
    $config['use_page_numbers'] = TRUE;

    $data['search_string'] = '';
    if(empty($search_string)) {
        $this->uri->segment(6, $this->uri->segment(5, 1));
        $data['search_string'] = $this->uri->segment(5, $search_string);
    } elseif($this->uri->segment(5) != null && empty($this->uri->segment(5)) && $this->uri->segment(6) != null) {
        $data['search_string'] = $this->uri->segment(5);
    }
    //set default page uri
    $page_uri = 5;

    if(empty($data['search_string']))
    $page_uri = 6;

    $config["uri_segment"] = $page_uri;

    $config["total_rows"] = $this->mahasiswa->record_count($data['search_string']);

    $data['page'] = $this->uri->segment($page_uri, 1);

    $data['sort_by'] = $this->uri->segment(3, 'NAMA_MHS');
    $orderBy = $this->uri->segment(4, "desc");
    $offset = ($data['page']-1) * $config['per_page'];
    $data['total_rows'] = $config["total_rows"];
    if($orderBy == "asc") $data['sort_order'] = "desc"; else $data['sort_order'] = "asc";

    $config["base_url"] = base_url(). 'Administrator/data_mahasiswa/'.$data['sort_by'].'/'.$orderBy.'/'.$data['search_string'];
    $config['full_tag_open'] = '<ul class="pagination">';
    $config['full_tag_close'] = '</ul>';
    $config['first_link'] = '&laquo; First';
    $config['first_tag_open'] = '<li class="prev page">';
    $config['first_tag_close'] = '</li>';

    $config['last_link'] = 'Last &raquo;';
    $config['last_tag_open'] = '<li class="next page">';
    $config['last_tag_close'] = '</li>';

    $config['next_link'] = 'Next &rarr;';
    $config['next_tag_open'] = '<li class="next page">';
    $config['next_tag_close'] = '</li>';

    $config['prev_link'] = '&larr; Previous';
    $config['prev_tag_open'] = '<li class="prev page">';
    $config['prev_tag_close'] = '</li>';

    $config['cur_tag_open'] = '<li class="active"><a href="">';
    $config['cur_tag_close'] = '</a></li>';

    $config['num_tag_open'] = '<li class="page">';
    $config['num_tag_close'] = '</li>';

    $data["data"] = $this->mahasiswa->get_mahasiswa($config["per_page"], $offset, $data['sort_by'], $data['sort_order'], $data['search_string']

    $this->pagination->initialize($config);
    $data["links"] = $this->pagination->create_links();

    $this->load->view('admin/v_datamahasiswa', $data);
}

```

Gambar 4. 25 Coding data mahasiswa

```

function get_mahasiswa($per_page, $offset, $sortfield, $orderBy, $search_string, $id=0)
{
    if(empty($id)){
        //echo $per_page.'fff'.$offset.'fff'.$sortfield.'fff'.$orderBy;
        if(!empty($search_string)) {
            $this->db->like('NAMA_MHS',$search_string);
            $this->db->or_like('NIM_MHS',$search_string);
            $this->db->or_like('PRODI_MHS',$search_string);
            $this->db->or_like('FAKULTAS_MHS',$search_string);
        }
        $this->db->order_by("$sortfield", "$orderBy");
        $this->db->limit($per_page,$offset);
        $query = $this->db->get('mahasiswa');
        if ($query->num_rows() > 0) {
            foreach ($query->result() as $row) {
                $data[] = $row;
            }
            return $data;
        }
        return false;
    } else {
        $query = $this->db->get_where('mahasiswa', array('NIM_MHS' => $id));
        return $query->row_array();
    }
}

public function record_count($search_string) {
    if(!empty($search_string)) {
        $this->db->like('NAMA_MHS',$search_string);
        $this->db->or_like('NIM_MHS',$search_string);
        $this->db->or_like('PRODI_MHS',$search_string);
        $this->db->or_like('FAKULTAS_MHS',$search_string);
    }
    return $this->db->count_all_results("mahasiswa");
}

```

Gambar 4. 26 Coding model data mahasiswa

Pada Gambar 4.27, dapat dilihat *coding* fungsi *edit_mhs* dalam *controller* Administrator memanggil fungsi *select* dalam *model* mahasiswa untuk mengambil data mahasiswa dengan parameter *nim_mhs*. Administrator dapat melakukan *reset password* mahasiswa melalui fungsi *profile_mhs* yang ada pada *model* mahasiswa. *Password* disimpan ke dalam *database* menggunakan enkripsi *md5*. Fungsi *select* dan *profile_mhs* dapat dilihat pada Gambar 4.28.

```

function edit_mhs($nim_mhs)
{
    if($_POST==NULL) {
        $data['data_mhs'] = $this->mahasiswa->select($nim_mhs);
        $this->load->view('admin/v_resetmahasiswa',$data);
    }else {
        $data['password_mhs'] = md5($this->input->post('password_mhs'));
        $nim_mhs = $this->input->post('nim_mhs');
        $this->mahasiswa->profile_mhs($nim_mhs, $data);
        redirect('Administrator/data_mahasiswa');
    }
}

```

Gambar 4. 27 Coding fungsi edit mahasiswa


```

function select($id_user)
{
    return $this->db->get_where('mahasiswa', array('nim_mhs'=>$id_user))->row();
}

//UPDATE
function profile_mhs($iduser, $data)
{
    $this->db->where(array('nim_mhs' => $iduser, 'nim_mhs !=' => 0));
    $this->db->update('mahasiswa', $data);
}

```

Gambar 4. 28 Coding fungsi *select* dan *profile_mhs*

Pada Gambar 4.29, dapat dilihat fungsi *delete_mhs* dalam *controller* Administrator yang memanggil fungsi *delete_mhs* dalam *model* mahasiswa. Fungsi *delete_mhs* dalam *model* mahasiswa dapat dilihat pada Gambar 4.30.

```

public function delete_mhs($id)
{
    $this->mahasiswa->delete_mhs($id);
    $this->session->set_flashdata('message', 'Deleted Successfully');

    redirect('Administrator/data_mahasiswa');
}

```

Gambar 4. 29 Coding fungsi *delete_mhs* dalam *controllers*

```

public function delete_mhs($id)
{
    $this->db->where('NIM_MHS', $id);
    return $this->db->delete('mahasiswa');
}

```

Gambar 4. 30 Coding fungsi *delete_mhs* dalam *models*

Pada Gambar 4.31 dapat dilihat fungsi aktivasi akun mahasiswa dalam *controller* Administrator memanggil fungsi *act_mhs* dalam *model* mahasiswa. Fungsi *act_mhs* dapat dilihat pada Gambar 4.32.

```

function aktivasi_mhs($nim_mhs)
{
    $this->mahasiswa->act_mhs($nim_mhs);
    $this->session->set_flashdata('message', 'Activated Successfully');

    redirect('Administrator/data_mahasiswa');
}

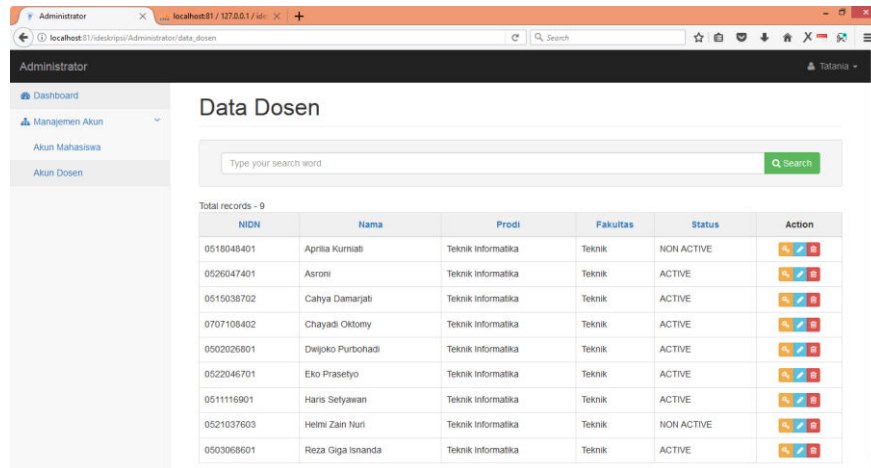
```

Gambar 4. 31 Coding aktivasi mahasiswa

```
function act_mhs($id)
{
    $this->db->set('STATUS_MHS', '1');
    $this->db->where('NIM_MHS', $id);
    $this->db->update('mahasiswa');
}
```

Gambar 4. 32 Fungsi *act_mhs*

Pada menu manajemen akun dosen, Administrator dapat melihat data akun dosen, melakukan pencarian, pengurutan data, aktivasi akun, mengubah *password*, dan menghapus akun dosen. Tampilan dapat dilihat pada Gambar 4.33, Gambar 4.33, Gambar 4.34, Gambar 4.35, Gambar 4.36, Gambar 4.37, Gambar 4.38.



Administrator

Dashboard

Manajemen Akun

Akun Mahasiswa

Akun Dosen

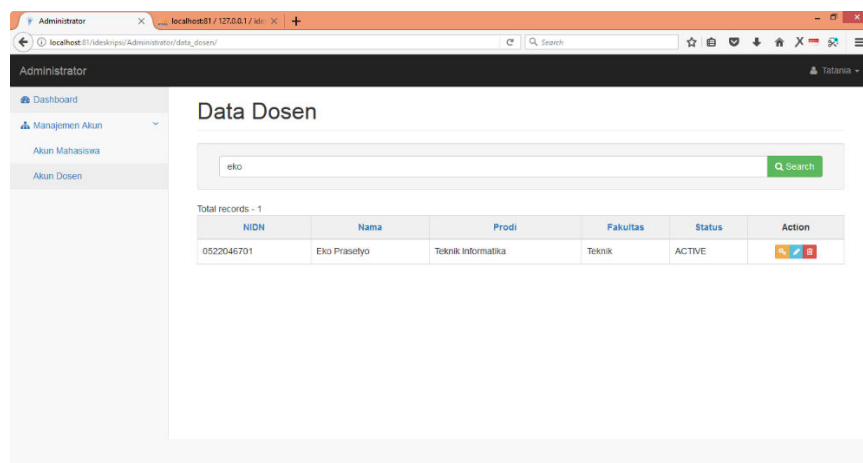
Data Dosen

Type your search word

Total records - 9

NIDN	Nama	Prodi	Fakultas	Status	Action
0518048401	Aprilia Kumati	Teknik Informatika	Teknik	NON ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0526047401	Asroni	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0515038702	Cahya Damarjati	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0707108402	Chayadi Oktomy	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0502026801	Dwijoko Purbohadi	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0522046701	Eko Prasetyo	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
051116901	Haris Setyawan	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0521037603	Helmi Zain Nuri	Teknik Informatika	Teknik	NON ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0503068601	Reza Giga Isnanda	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>

Gambar 4. 33 Tampilan manajemen akun dosen



Administrator

Dashboard

Manajemen Akun

Akun Mahasiswa

Akun Dosen

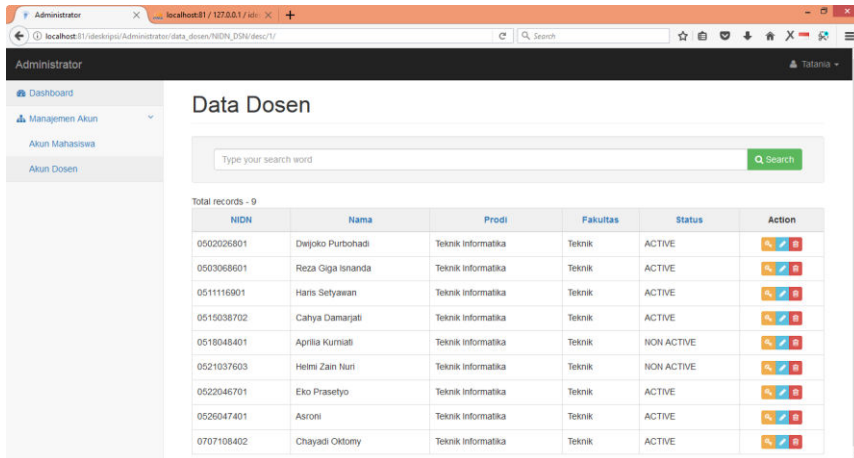
Data Dosen

eko

Total records - 1

NIDN	Nama	Prodi	Fakultas	Status	Action
0522046701	Eko Prasetyo	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>

Gambar 4. 34 Tampilan pencarian data dosen



Administrator

Dashboard

Manajemen Akun

Akun Mahasiswa

Akun Dosen

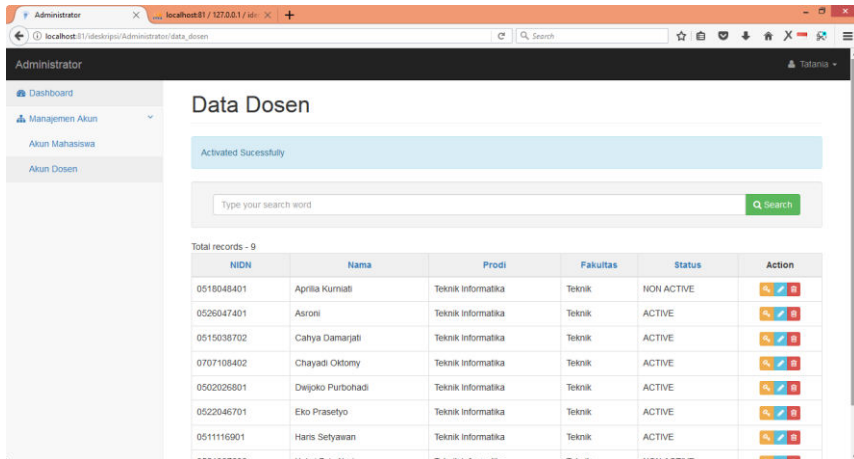
Data Dosen

Type your search word

Total records - 9

NIDN	Nama	Prodi	Fakultas	Status	Action
0502026801	Dwijoko Purbohadi	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0503068601	Reza Giga Isnanda	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0511116901	Haris Setyawan	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0515038702	Cahya Damarjati	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0518048401	Aprilia Kumali	Teknik Informatika	Teknik	NON ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0521037603	Helmi Zain Nuri	Teknik Informatika	Teknik	NON ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0522046701	Eko Prasetyo	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0526047401	Asroni	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0707108402	Chayadi Oktomy	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>

Gambar 4. 35 Tampilan data dosenurut berdasarkan NIDN



Administrator

Dashboard

Manajemen Akun

Akun Mahasiswa

Akun Dosen

Data Dosen

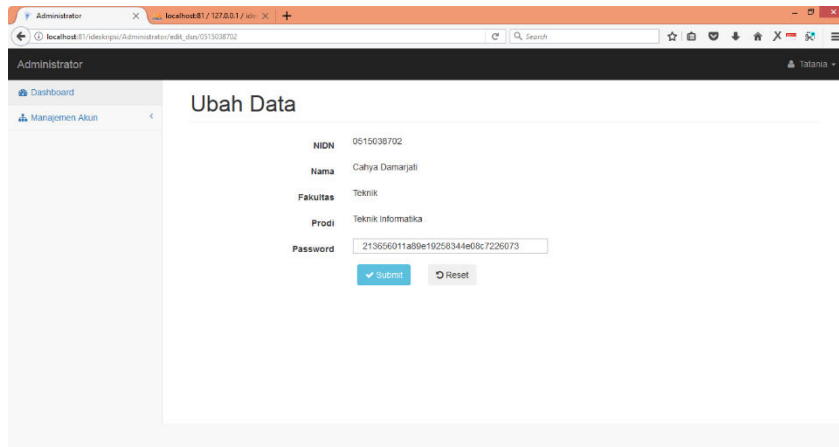
Activated Successfully

Type your search word

Total records - 9

NIDN	Nama	Prodi	Fakultas	Status	Action
0518048401	Aprilia Kumali	Teknik Informatika	Teknik	NON ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0526047401	Asroni	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0515038702	Cahya Damarjati	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0707108402	Chayadi Oktomy	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0502026801	Dwijoko Purbohadi	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0522046701	Eko Prasetyo	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0511116901	Haris Setyawan	Teknik Informatika	Teknik	ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>
0521037603	Helmi Zain Nuri	Teknik Informatika	Teknik	NON ACTIVE	<input type="button" value="edit"/> <input type="button" value="delete"/>

Gambar 4. 36 Tampilan berhasil aktivasi akun dosen



Administrator

Dashboard

Manajemen Akun

Ubah Data

NIDN 0515038702

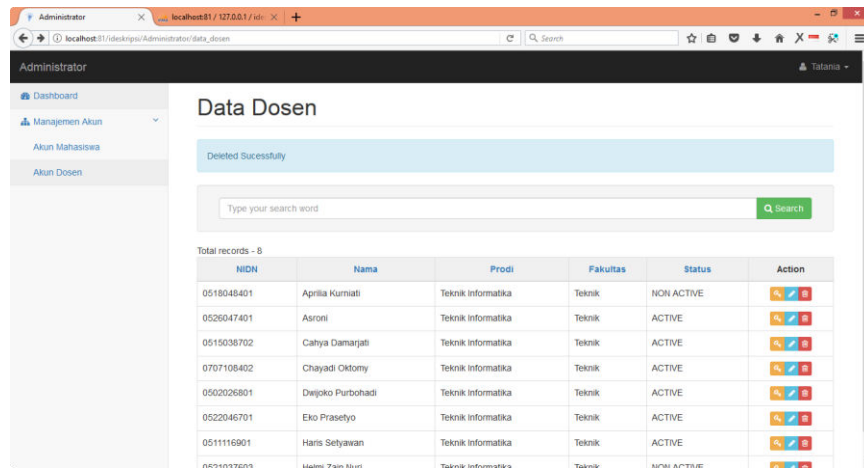
Nama Cahya Damarjati

Fakultas Teknik

Prodi Teknik Informatika

Password

Gambar 4. 37 Tampilan ubah password akun dosen



Gambar 4.38 Tampilan berhasil menghapus akun dosen

Pada Gambar 4.39, dapat dilihat bahwa fungsi `data_dosen` memanggil fungsi `get_dosen` dan `record_count` yang ada pada `M_Dsn` dalam folder `models`. Fungsi `get_dosen` digunakan untuk mendapatkan data dosen dari `database`. Sedangkan fungsi `record_count` digunakan untuk menghitung jumlah data dosen. `Coding` fungsi `get_dosen` dan `record_count` dapat dilihat pada Gambar 4.40.

```

public function data_dosen()
{
    $data = array();
    $data['sort_cols'] = array(
        'NIDN_DSN' => 'NIDN',
        'NAMA_DSN' => 'Nama',
        'PRODI_DSN' => 'Prodi',
        'FAKULTAS_DSN' => 'Fakultas'
    );
    $config = array();
    //base_url () . 'index.php/questions/page/'. $sortfield.'/'. $order.'/',
    $search_string = $this->input->post('search');

    $config["per_page"] = 10;
    //max number of page links
    $config['num_links'] = 2;
    //use page number as parameter
    $config['use_page_numbers'] = TRUE;

    $data['search_string'] = '';
    if(!empty($search_string)) {
        $this->uri->segment(6, $this->uri->segment(5, 1));
        $data['search_string'] = $this->uri->segment(5, $search_string);
    } elseif($this->uri->segment(5) != null && !empty($this->uri->segment(5)) && $this->uri->segment(6) != null) {
        $data['search_string'] = $this->uri->segment(5);
    }
    //set default page uri
    $page_uri = 5;

    if(!empty($data['search_string']))
        $page_uri = 6;

    $config["uri_segment"] = $page_uri;

    $config["total_rows"] = $this->dosen->record_count($data['search_string']);

    $data['page'] = $this->uri->segment($page_uri, 1);

    $data['sort_by'] = $this->uri->segment(3, 'NAMA_DSN');
    $order_by = $this->uri->segment(4, "desc");
    $offset = ($data['page']-1) * $config['per_page'];
    $data['total_rows'] = $config["total_rows"];
    if($order_by == "asc") $data['sort_order'] = "desc"; else $data['sort_order'] = "asc";

    $config["base_url"] = base_url(). 'Administrator/data_dosen/'. $data['sort_by'].'/'. $order_by.'/'. $data['search_string'];
    $config['full_tag_open'] = '<ul class="pagination">';
    $config['full_tag_close'] = '</ul>';
    $config['first_link'] = '&laquo; First';
    $config['first_tag_open'] = '<li class="prev page">';
    $config['first_tag_close'] = '</li>';

    $config['last_link'] = 'Last &raquo;';
    $config['last_tag_open'] = '<li class="next page">';
    $config['last_tag_close'] = '</li>';

    $config['next_link'] = 'Next &rarr;';
    $config['next_tag_open'] = '<li class="next page">';
    $config['next_tag_close'] = '</li>';

    $config['prev_link'] = '&larr; Previous';
    $config['prev_tag_open'] = '<li class="prev page">';
    $config['prev_tag_close'] = '</li>';

    $config['cur_tag_open'] = '<li class="active"><a href="">';
    $config['cur_tag_close'] = '</a></li>';

    $config['num_tag_open'] = '<li class="page">';
    $config['num_tag_close'] = '</li>';

    $data["data"] = $this->dosen->get_dosen($config["per_page"], $offset, $data['sort_by'], $data['sort_order'], $data['search_string']);

    $this->pagination->initialize($config);
    $data["links"] = $this->pagination->create_links();

    $this->load->view('admin/v_datadosen', $data);
}

```

Gambar 4. 39 Coding data dosen

```

function get_dosen($per_page, $offset, $sortfield, $orderBy, $search_string, $id=0)
{
    if(empty($id)){
        //echo $per_page.'fff'.$offset.'fff'.$sortfield.'fff'.$orderBy;
        if(!empty($search_string)) {
            $this->db->like('NAMA_DSN',$search_string);
            $this->db->or_like('NIDN_DSN',$search_string);
            $this->db->or_like('PRODI_DSN',$search_string);
            $this->db->or_like('FAKULTAS_DSN',$search_string);
        }
        $this->db->order_by("$sortfield", "$orderBy");
        $this->db->limit($per_page,$offset);
        $query = $this->db->get('dosen');
        if ($query->num_rows() > 0) {
            foreach ($query->result() as $row) {
                $data[] = $row;
            }
        }
        return $data;
    }
    return false;
} else {
    $query = $this->db->get_where('dosen', array('NIDN_DSN' => $id));
    return $query->row_array();
}
}

public function record_count($search_string) {
    if(!empty($search_string)) {
        $this->db->like('NAMA_DSN',$search_string);
        $this->db->or_like('NIDN_DSN',$search_string);
        $this->db->or_like('PRODI_DSN',$search_string);
        $this->db->or_like('FAKULTAS_DSN',$search_string);
    }
    return $this->db->count_all_results("dosen");
}
}

```

Gambar 4. 40 Coding model data mahasiswa

Pada Gambar 4.41, dapat dilihat *coding* fungsi *edit_dsn* dalam *controller* Administrator memanggil fungsi *select* dalam *model* dosen untuk mengambil data dosen dengan parameter *nidn_dsn*. Administrator dapat melakukan *reset password* dosen melalui fungsi *profile_dsn* yang ada pada *model* dosen. *Password* disimpan ke dalam *database* menggunakan enkripsi md5. Fungsi *select* dan *profile_dsn* dapat dilihat pada Gambar 4.42.

```

function edit_dsn($nidn_dsn)
{
    if($_POST==NULL) {
        $data['data_dsn'] = $this->dosen->select($nidn_dsn);
        $this->load->view('admin/v_resetdosen',$data);
    }else {
        $data['password_dsn'] = md5($this->input->post('password_dsn'));
        $nidn_dsn = $this->input->post('nidn_dsn');
        $this->dosen->profile_dsn($nidn_dsn, $data);
        redirect('Administrator/data_dosen');
    }
}
}

```

Gambar 4. 41 Coding fungsi edit dosen

```

function select($id_user)
{
    return $this->db->get_where('dosen', array('NIDN_DSN'=>$id_user))->row();
}

//UPDATE
function profile_dsn($iduser, $data)
{
    $this->db->where(array('NIDN_DSN' => $iduser, 'NIDN_DSN !=' => 0));
    $this->db->update('dosen', $data);
}

```

Gambar 4. 42 Coding fungsi *select* dan *profile_dsn*

Pada Gambar 4.43, dapat dilihat fungsi *delete_dsn* dalam *controller* Administrator yang memanggil fungsi *delete_dsn* dalam *model* dosen. Fungsi *delete_dsn* dalam *model* dosen dapat dilihat pada Gambar 4.44.

```

public function delete_dsn($id)
{
    $this->dosen->delete_dsn($id);
    $this->session->set_flashdata('message', 'Deleted Sucessfully');
    redirect('Administrator/data_dosen');
}

```

Gambar 4. 43 Coding fungsi *delete_dsn* dalam *controllers*

```

public function delete_dsn($id)
{
    $this->db->where('NIDN_DSN', $id);
    return $this->db->delete('dosen');
}

```

Gambar 4. 44 Coding fungsi *delete_dsn* dalam *models*

Pada Gambar 4.45, dapat dilihat fungsi aktivasi akun dosen dalam *controller* Administrator yang memanggil fungsi *act_dsn* dalam *model* dosen. Fungsi *act_dsn* dalam *model* dosen dapat dilihat pada Gambar 4.46.

```

function aktivasi_dsn($nidn_dsn)
{
    $this->dosen->act_dsn($nidn_dsn);
    $this->session->set_flashdata('message', 'Activated Sucessfully');
    redirect('Administrator/data_dosen');
}

```

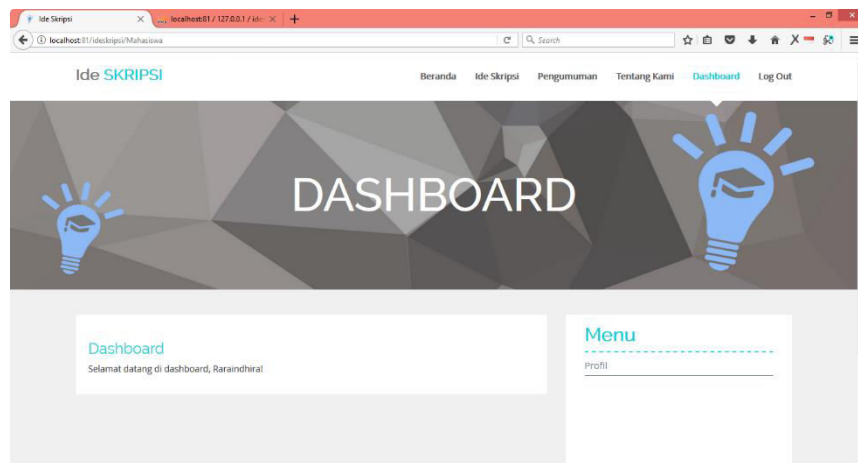
Gambar 4. 45 Coding aktivasi akun dosen

```
function act_dsn($id)
{
    $this->db->set('STATUS_DSN', '1');
    $this->db->where('NIDN_DSN', $id);
    $this->db->update('dosen');
}
```

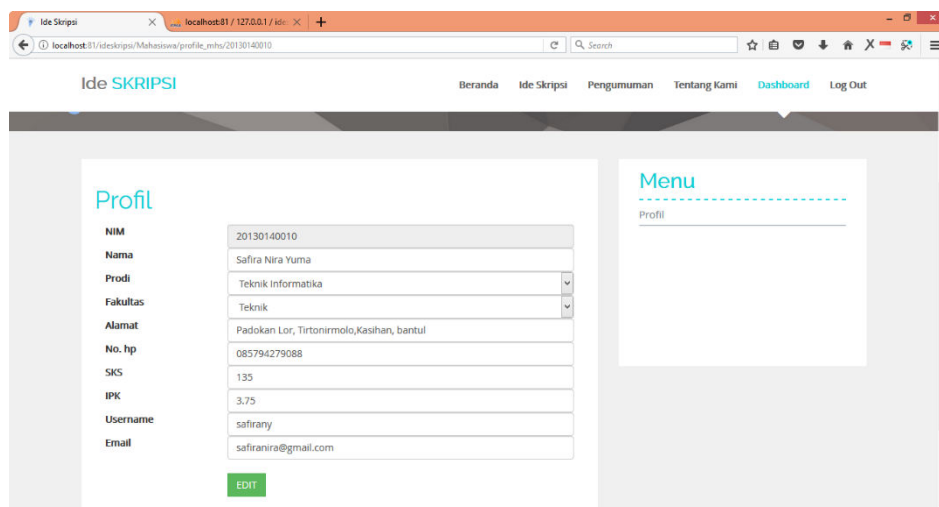
Gambar 4. 46 Fungsi *act_dsn*

4.2.6 Halaman *Dashboard* Mahasiswa

Halaman *dashboard* mahasiswa adalah halaman utama untuk mahasiswa setelah *login*. Pada halaman ini terdapat menu profil, dimana mahasiswa dapat mengubah data diri. Tampilan dapat dilihat pada Gambar 4.41 dan 4.42.



Gambar 4. 47 Tampilan *dashboard* mahasiswa



Gambar 4. 48 Tampilan ubah profil mahasiswa

Pada Gambar 4.43, dapat dilihat fungsi untuk mengubah profil mahasiswa dalam *controller* mahasiswa. Jika validasi benar, data mahasiswa dalam *database* akan diperbarui.

```

public function profile_mhs($id_user)
{
    //GET REQUIRED DATA FROM DB
    $data['data_mhs'] = $this->M_Mhs->select($id_user);
    $this->load->view('mhs/Vm_mhs-profile', $data);
}
public function profile_mhs_update()
{
    $this->form_validation->set_rules('nama', 'NAMA MAHASISWA', 'required');
    $this->form_validation->set_rules('prodi', 'PRODI', 'required');
    $this->form_validation->set_rules('fakultas', 'FAKULTAS', 'required');
    $this->form_validation->set_rules('alamat', 'ALAMAT', 'required');
    $this->form_validation->set_rules('nohp', 'NO HP', 'required');
    $this->form_validation->set_rules('sks', 'SKS', 'required');
    $this->form_validation->set_rules('ipk', 'IPK', 'required');
    $this->form_validation->set_rules('username', 'USERNAME', 'required');
    $this->form_validation->set_rules('email', 'EMAIL', 'required');

    if($this->form_validation->run() == FALSE)
    {
        $this->load->view('mhs/Vm_mhs-profile');
    }
    else
    {
        $data['nama_mhs'] = $this->input->post('nama');
        $data['prodi_mhs'] = $this->input->post('prodi');
        $data['fakultas_mhs'] = $this->input->post('fakultas');
        $data['alamat_mhs'] = $this->input->post('alamat');
        $data['nohp_mhs'] = $this->input->post('nohp');
        $data['sks_mhs'] = $this->input->post('sks');
        $data['ipk_mhs'] = $this->input->post('ipk');
        $data['username_mhs'] = $this->input->post('username');
        $data['email_mhs'] = $this->input->post('email');

        $id_mhs = $this->input->post('nim');
        $this->M_Mhs->profile_mhs($id_mhs, $data);
        redirect(site_url('Mahasiswa'));
    }
}

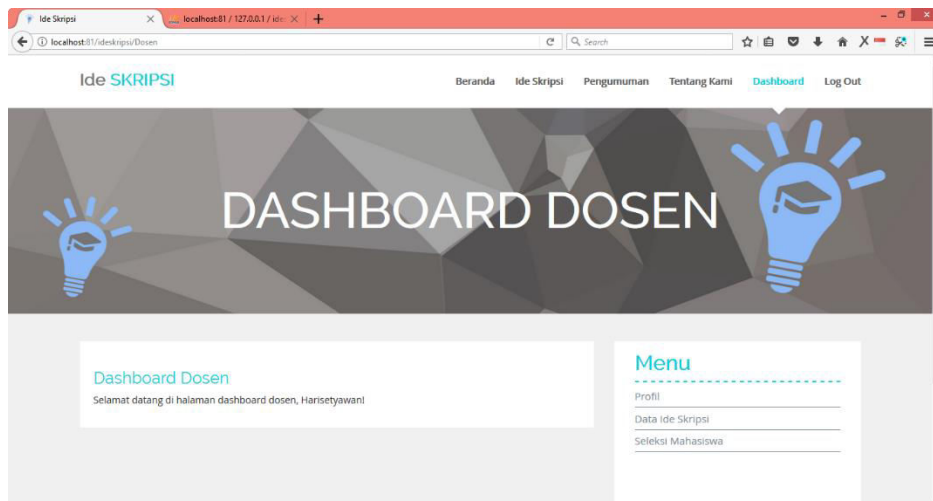
```

Gambar 4. 49 Coding ubah profil mahasiswa

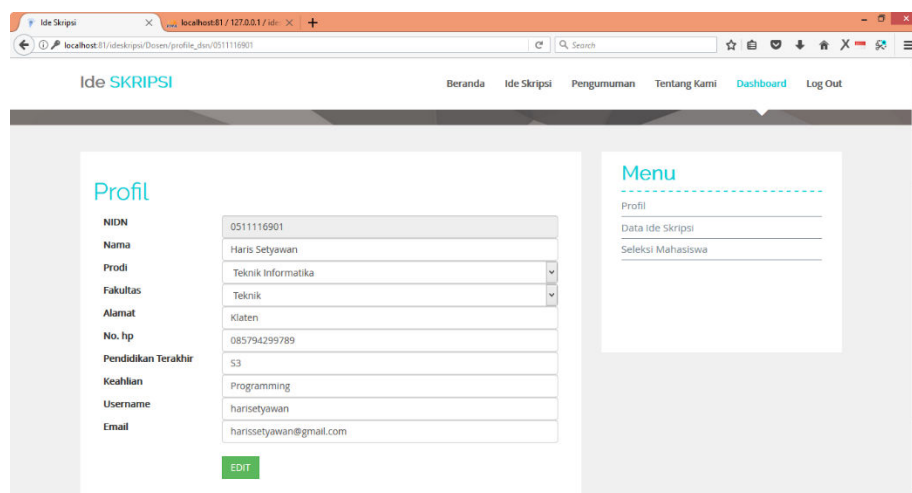
4.2.7 Halaman *Dashboard* Dosen

Halaman *dashboard* dosen adalah halaman utama untuk dosen setelah *login*. Pada halaman ini terdapat beberapa menu, meliputi profil, data ide skripsi, dan seleksi mahasiswa. Pada menu profil, dosen dapat mengubah data diri. Menu data ide skripsi menampilkan data ide skripsi dosen yang bersangkutan. Pada menu ide skripsi, dosen dapat melakukan pencarian, pengurutan, penambahan, pengubahan, dan penghapusan data ide skripsi. Menu seleksi mahasiswa menampilkan data mahasiswa yang mengambil ide skripsi. Pada menu seleksi mahasiswa, dosen dapat

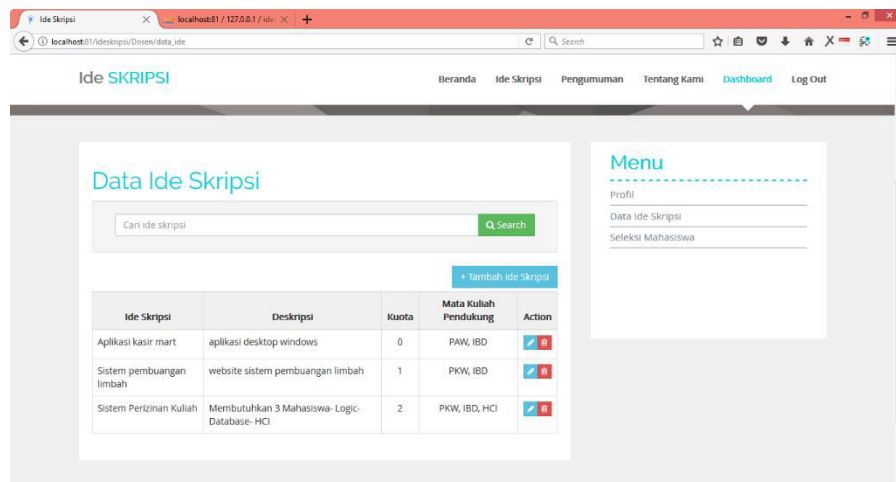
memilih mahasiswa untuk kemudian memberikan keputusan diterima atau tidak diterima. Tampilan dapat dilihat pada Gambar 4.50, Gambar 4.51, dan Gambar 4.52, Gambar 4.53, Gambar 4.54, Gambar 4.55, Gambar 4.56, Gambar 4.57, Gambar 4.58, Gambar 4.59, Gambar 4.60.



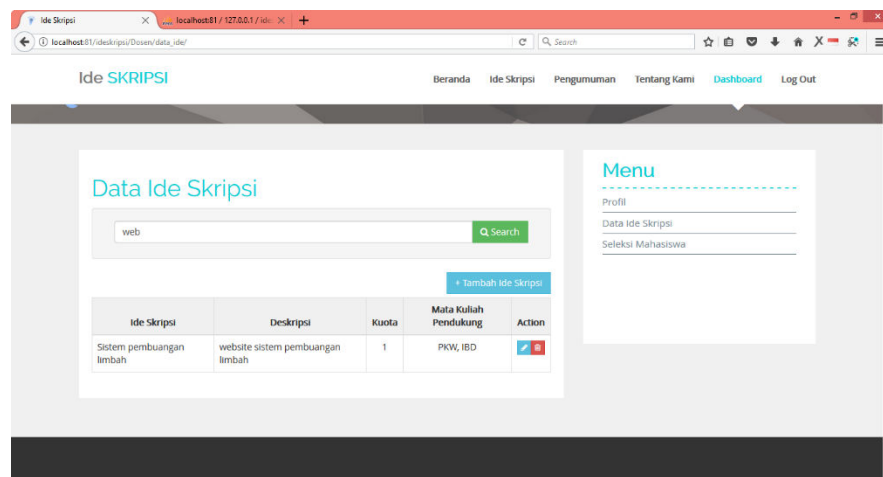
Gambar 4. 50 Tampilan *dashboard* dosen



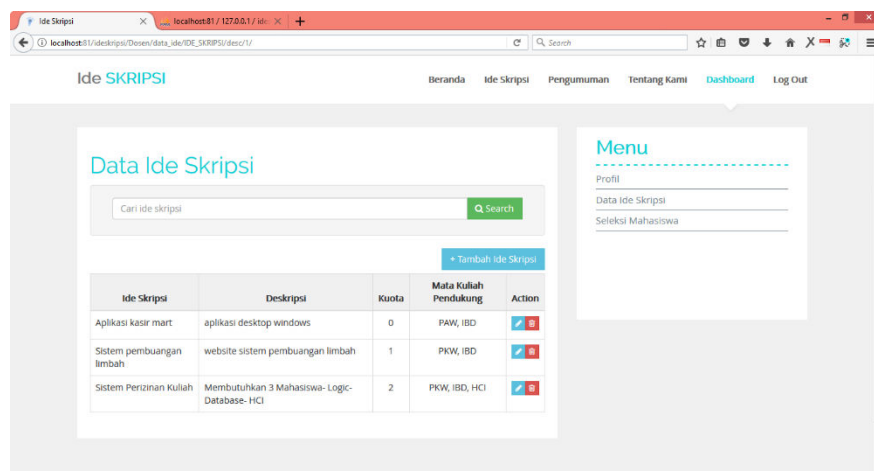
Gambar 4. 51 Tampilan profil dosen



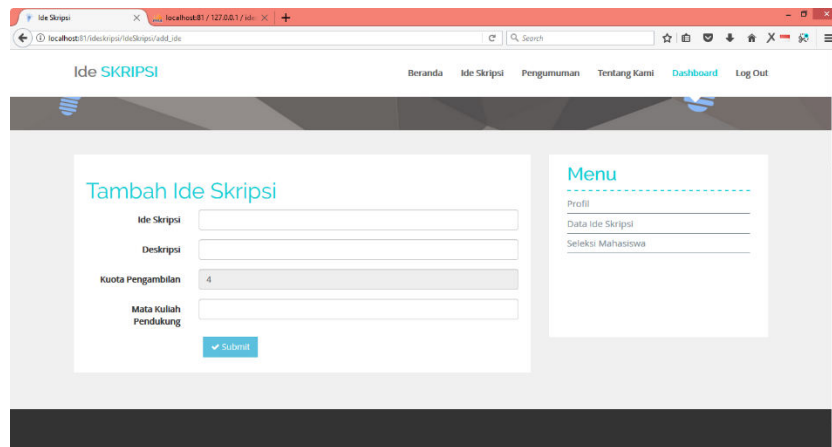
Gambar 4. 52 Tampilan menu data ide skripsi



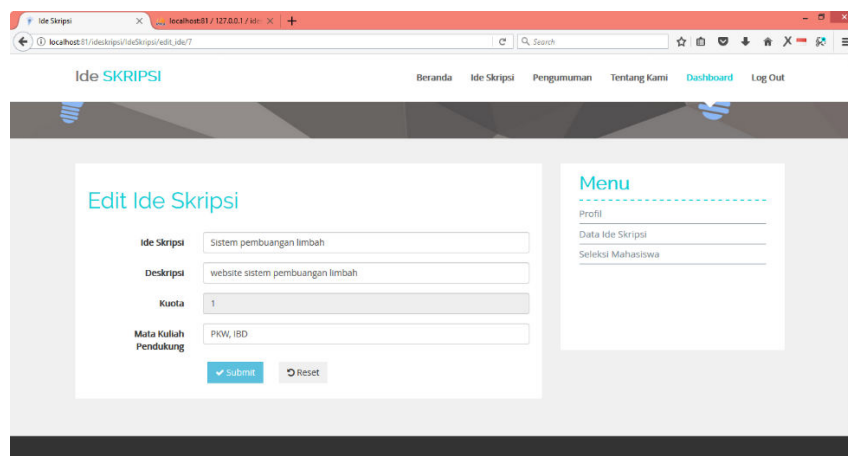
Gambar 4. 53 Tampilan pencarian ide skripsi dosen



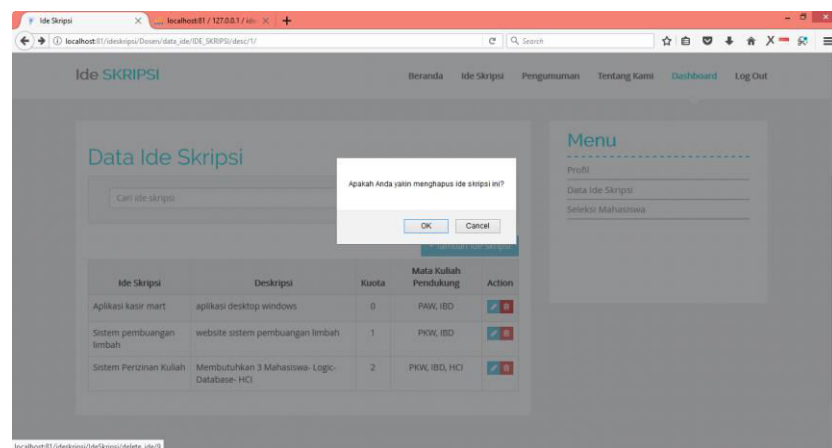
Gambar 4. 54 Tampilan data ide skripsi dosen urut judul



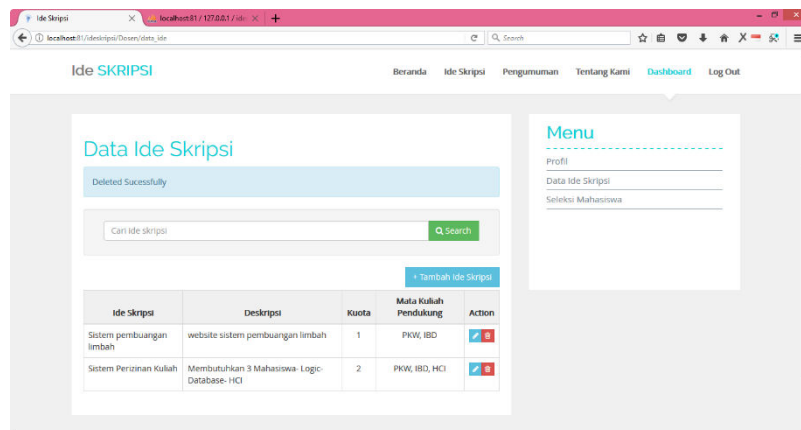
Gambar 4. 55 Tampilan tambah ide skripsi



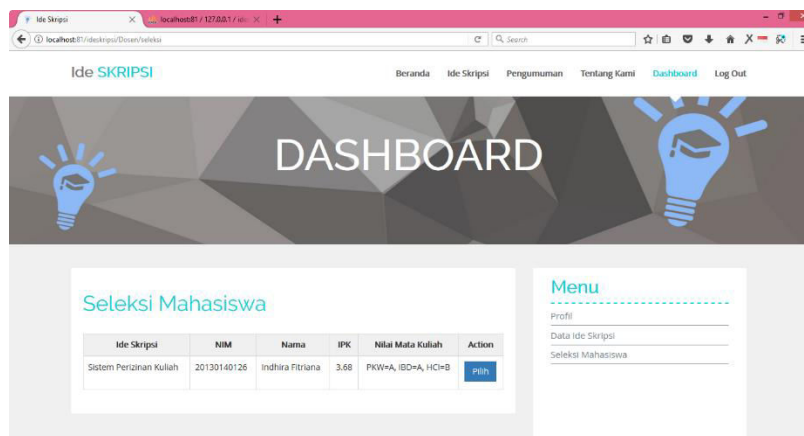
Gambar 4. 56 Tampilan ubah ide skripsi



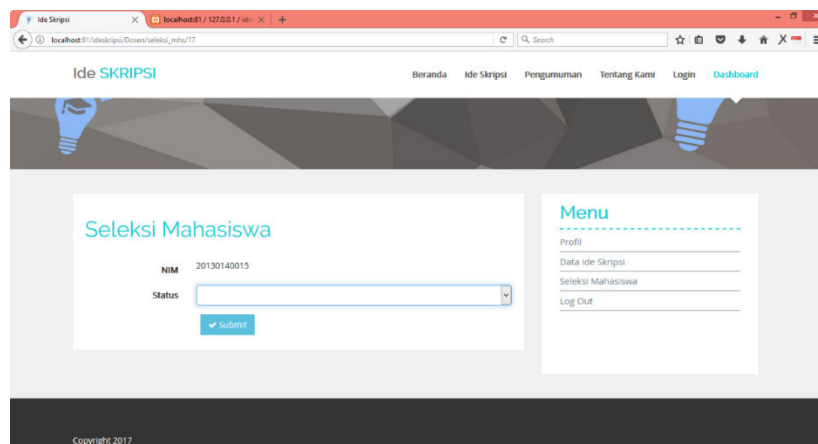
Gambar 4. 57 Tampilan menghapus ide skripsi



Gambar 4. 58 Tampilan berhasil menghapus ide skripsi



Gambar 4. 59 Tampilan menu seleksi mahasiswa



Gambar 4. 60 Tampilan seleksi mahasiswa

Pada Gambar 4.61, dapat dilihat fungsi untuk mengubah profil dosen dalam *controller* dosen. Jika validasi benar, data dosen dalam *database* akan diperbarui.

```

public function profile_dsn($id_user)
{
    //GET REQUIRED DATA FROM DB
    $data['data_dsn'] = $this->M_Dsn->select($id_user);
    $this->load->view('dsn/Vm_dsn-profile', $data);
}
public function profile_dsn_update()
{
    $this->form_validation->set_rules('nama','NAMA MAHASISWA','required');
    $this->form_validation->set_rules('prodi','PRODI','required');
    $this->form_validation->set_rules('fakultas','FAKULTAS','required');
    $this->form_validation->set_rules('alamat','ALAMAT','required');
    $this->form_validation->set_rules('nohp','NO HP','required');
    $this->form_validation->set_rules('pendidikanterakhir','PENDIDIKAN TERAKHIR','required');
    $this->form_validation->set_rules('keahlian','KEAHLIAN','required');
    $this->form_validation->set_rules('username','USERNAME','required');
    $this->form_validation->set_rules('email','EMAIL','required');

    if($this->form_validation->run() == FALSE)
    {
        $this->load->view('dsn/Vm_dsn-profile');
    }
    else
    {
        $data['nama_dsn'] = $this->input->post('nama');
        $data['prodi_dsn'] = $this->input->post('prodi');
        $data['fakultas_dsn'] = $this->input->post('fakultas');
        $data['alamat_dsn'] = $this->input->post('alamat');
        $data['nohp_dsn'] = $this->input->post('nohp');
        $data['pendidikan_dsn'] = $this->input->post('pendidikanterakhir');
        $data['keahlian_dsn'] = $this->input->post('keahlian');
        $data['username_dsn'] = $this->input->post('username');
        $data['email_dsn'] = $this->input->post('email');

        $id_dsn = $this->input->post('nidn');
        $this->M_Dsn->profile_dsn($id_dsn, $data);
        redirect(site_url('Dosen'));
    }
}

```

Gambar 4. 61 Coding ubah profil dosen

Pada Gambar 4.62, dapat dilihat bahwa fungsi *data_ide* memanggil fungsi *select_ide* yang ada pada *model ide*. Fungsi *select_ide* digunakan untuk mendapatkan data ide berdasarkan NIDN dosen dari *database*. Coding fungsi *select_ide* dapat dilihat pada Gambar 4.63.

```

public function data_ide()
{
    //GET REQUIRED DATA FROM DB
    $data = array();
    $data['title'] = 'Ide Skripsi';
    $data['sort_cols'] = array(
        'IDE_SKRIPSI' => 'Ide Skripsi',
        'DESKRIPSI' => 'Deskripsi',
        'KUOTA_AMBIL' => 'Kuota',
        'MATKUL' => 'Mata Kuliah Pendukung'
    );
    $config = array();
    //base_url () . 'index.php/questions/page/'. $sortfield.'/'. $order.'/',
    $search_string = $this->input->post('search');

    $config['per_page'] = 10;
    //max number of page links
    $config['num_links'] = 2;
    //use page number as parameter
    $config['use_page_numbers'] = TRUE;

    $data['search_string'] = '';
    if(empty($search_string)) {
        $this->uri->segment(6, $this->uri->segment(5, 1));
        $data['search_string'] = $this->uri->segment(5, $search_string);
    } elseif($this->uri->segment(5) != null && empty($this->uri->segment(5)) && $this->uri->segment(6) != null {
        $data['search_string'] = $this->uri->segment(5);
    }
    //set default page uri
    $page_uri = 5;

    if(empty($data['search_string']))
        $page_uri = 6;

    $config['uri_segment'] = $page_uri;

    $config['total_rows'] = $this->ide->record_count($data['search_string']);

    $data['page'] = $this->uri->segment($page_uri, 1);

    $data['sort_by'] = $this->uri->segment(3, 'IDE_SKRIPSI');
    $order_by = $this->uri->segment(4, "desc");
    $offset = ($data['page']-1) * $config['per_page'];
    $data['total_rows'] = $config['total_rows'];
    if($order_by == "asc") $data['sort_order'] = "desc"; else $data['sort_order'] = "asc";

    $config['base_url'] = base_url(). 'Dosen/data_ide/'. $data['sort_by'].'/'. $order_by.'/'. $data['search_string'];
    $config['full_tag_open'] = '<ul class="pagination">';
    $config['full_tag_close'] = '</ul>';
    $config['first_link'] = '&laquo; First';
    $config['first_tag_open'] = '<li class="prev page">';
    $config['first_tag_close'] = '</li>';

    $config['last_link'] = 'Last &raquo;';
    $config['last_tag_open'] = '<li class="next page">';
    $config['last_tag_close'] = '</li>';

    $config['next_link'] = 'Next &rarr;';
    $config['next_tag_open'] = '<li class="next page">';
    $config['next_tag_close'] = '</li>';

    $config['prev_link'] = '&larr; Previous';
    $config['prev_tag_open'] = '<li class="prev page">';
    $config['prev_tag_close'] = '</li>';

    $config['cur_tag_open'] = '<li class="active"><a href="">';
    $config['cur_tag_close'] = '</a></li>';

    $config['num_tag_open'] = '<li class="page">';
    $config['num_tag_close'] = '</li>';

    $data["data"] = $this->ide->select_ide($config["per_page"], $offset, $data["sort_by"], $data["sort_order"], $data["search_string"]);

    $this->pagination->initialize($config);
    $data["links"] = $this->pagination->create_links();

    $this->load->view('dsn/v_idedosen', $data);
}

```

Gambar 4. 62 Coding data ide dosen

```

function select_ide($per_page, $offset, $sortfield, $orderBy, $search_string, $id=0)
{
    if(empty($id)){
        //echo $per_page.'fff'.$offset.'fff'.$sortfield.'fff'.$orderBy;
        if(!empty($search_string)) {

            $this->db->like('IDE_SKRIPSI',$search_string);
            $this->db->or_like('DESKRIPSI',$search_string);
        }
        $this->db->order_by("$sortfield", "$orderBy");
        $this->db->limit($per_page,$offset);
        $this->db->select("ID_IDE, IDE_SKRIPSI, DESKRIPSI, KUOTA_AMBIL, NIDN_DSN, MATKUL");
        $this->db->from('ideskripsi');
        $this->db->where('NIDN_DSN', $this->session->userdata('id_user'));
        $query = $this->db->get();
        if ($query->num_rows() > 0) {
            foreach ($query->result() as $row) {
                $data[] = $row;
            }
            return $data;
        }
        return false;
    } else {
        $query = $this->db->get_where('ideskripsi', array('ID_IDE' => $id));
        return $query->row_array();
    }
}

```

Gambar 4. 63 Coding select ide dosen

Pada Gambar 4.64, dapat dilihat bahwa fungsi *add_ide* pada *controller* IdeSkripsi memanggil fungsi *add_ide* pada *model* ide. Fungsi *add_ide* pada *model* ide dapat dilihat pada Gambar 4.65.

```

public function add_ide()
{
    $this->form_validation->set_rules('ideskripsi', 'IDE SKRIPSI', 'required|is_unique[ideskripsi.IDE_SKRIPSI][max_length[60]',
        array(
            'is_unique' => 'This %s already exists.'));
    $this->form_validation->set_rules('deskripsi', 'DESKRIPSI', 'required');
    $this->form_validation->set_rules('kuota_ambil', 'KUOTA AMBIL', 'required');
    $this->form_validation->set_rules('matkul', 'MATKUL PENDUKUNG', 'required');

    if($this->form_validation->run() == FALSE) {
        $this->load->view('dsn/v_tambahide');
    }else{

        $data['IDE_SKRIPSI'] = $this->input->post('ideskripsi');
        $data['DESKRIPSI'] = $this->input->post('deskripsi');
        $data['KUOTA_AMBIL'] = $this->input->post('kuota_ambil');
        $data['MATKUL'] = $this->input->post('matkul');
        $data['NIDN_DSN'] = $this->input->post('nidn');

        $this->ide->add_ide($data);
        redirect('Dosen/data_ide');
    }
}

```

Gambar 4. 64 Coding tambah ide

```

function add_ide($data)
{
    $this->db->insert('ideskripsi',$data);
}

```

Gambar 4. 65 Fungsi tambah ide

Pada Gambar 4.66, dapat dilihat *coding* fungsi *edit_ide* dalam *controller* IdeSkripsi memanggil fungsi *select* dalam *model* ide untuk mengambil data ide dengan parameter id ide. Dosen dapat mengubah data ide skripsi melalui fungsi

update_ide yang ada pada *model ide*. Fungsi *select* dan *update_ide* dapat dilihat pada Gambar 4.67.

```
function edit_ide($id)
{
    if($_POST==NULL) {
        $data['data_ide'] = $this->ide->select($id);
        $this->load->view('dsn/v_editide',$data);
    }else {
        $data['IDE_SKRIPSI'] = $this->input->post('ideskripsi');
        $data['DESKRIPSI'] = $this->input->post('deskripsi');
        $data['KUOTA_AMBIL'] = $this->input->post('kuota_ambil');
        $data['MATKUL'] = $this->input->post('matkul');

        $id = $this->input->post('id');
        $this->ide->update_ide($id, $data);
        redirect('Dosen/data_ide');
    }
}
```

Gambar 4. 66 Coding edit ide

```
function select($id)
{
    return $this->db->get_where('ideskripsi', array('ID_IDE'=>$id))->row();
}

//UPDATE
function update_ide($id, $data)
{
    $this->db->where(array('ID_IDE' => $id, 'ID_IDE !=' => 0));
    $this->db->update('ideskripsi', $data);
}
```

Gambar 4. 67 Fungsi *select* dan *update ide*

Pada Gambar 4.68, dapat dilihat *coding* fungsi *delete_ide* dalam *controller IdeSkripsi* memanggil fungsi *delete_ide* dalam *model ide* untuk menghapus data ide dengan parameter id ide. Fungsi *delete_ide* pada *model ide* dapat dilihat pada Gambar 4.69.

```
public function delete_ide($id)
{
    $this->ide->delete_ide($id);
    $this->session->set_flashdata('message', 'Deleted Sucessfully');

    redirect('Dosen/data_ide');
}
```

Gambar 4. 68 Coding hapus ide skripsi

```
public function delete_ide($id)
{
    $this->db->where('ID_IDE', $id);
    return $this->db->delete('ideskripsi');
}
```

Gambar 4. 69 Fungsi *delete ide*

Pada Gambar 4.70, dapat dilihat bahwa fungsi seleksi memanggil fungsi *get_pengambilan* yang ada pada *model* pengambilan. Fungsi *get_pengambilan* digunakan untuk mendapatkan data mahasiswa pengambil ide skripsi berdasarkan NIDN dosen yang mempunyai ide skripsi dari *database*. *Coding* fungsi *get_pengambilan* dapat dilihat pada Gambar 4.71.

```

public function seleksi()
{
    //GET REQUIRED DATA FROM DB
    $data = array();
    $data['title'] = 'Seleksi';
    $data['sort_cols'] = array(
        'IDE_SKRIPSI' => 'Ide Skripsi',
        'NIM_MHS' => 'NIM',
        'NAMA_MHS' => 'Nama',
        'IPK_MHS' => 'IPK',
        'NILAI_MATKUL' => 'Nilai Mata Kuliah'
    );
    $config = array();

    $data['sort_by'] = $this->uri->segment(3, 'IDE_SKRIPSI');
    $orderBy = $this->uri->segment(4, "desc");

    if($orderBy == "asc") $data['sort_order'] = "desc"; else $data['sort_order'] = "asc";

    $data["data"] = $this->pengambilan->get_pengambilan($data['sort_by'], $data['sort_order']);

    $this->load->view('dsn/v_seleksi', $data);
}

```

Gambar 4. 70 *Coding* data seleksi

```

function get_pengambilan($sortfield, $orderBy, $id=0)
{
    if(empty($id)){
        $this->db->order_by("$sortfield", "$orderBy");
        $this->db->select("pengambilanide.ID_PIS, pengambilanide.NIM_MHS, pengambilanide.NILAI_MATKUL, ideskripsi.IDE_SKRIPSI, mahasiswa.NAMA_MH");
        $this->db->from('pengambilanide');
        $this->db->join('ideskripsi', 'ideskripsi.ID_IDE = pengambilanide.ID_IDE');
        $this->db->join('mahasiswa', 'mahasiswa.NIM_MHS = pengambilanide.NIM_MHS');
        $array = array('NIDN_DSN' => $this->session->userdata('id_user'), 'STATUS' => NULL);
        $this->db->where($array);
        $query = $this->db->get();
        if ($query->num_rows() > 0) {
            foreach ($query->result() as $row) {
                $data[] = $row;
            }
        }
        return $data;
    }
    return false;
} else {
    $query = $this->db->get_where('pengambilanide', array('ID_AD' => $id));
    return $query->row_array();
}
}

```

Gambar 4. 71 Fungsi *get_pengambilan*

Pada Gambar 4.72, dapat dilihat bahwa fungsi *seleksi_mhs* memanggil fungsi *select* dan *update_status* pada *model* pengambilan. Fungsi *select* digunakan untuk mengambil data pengambil ide skripsi. Sedangkan, fungsi *update_status* untuk mengubah status pengambil ide skripsi. *Coding* fungsi *select* dan *update_status* pada *model* pengambilan dapat dilihat pada Gambar 4.73.

```

public function seleksi_mhs($id_p)
{
    if($_POST==NULL) {
        $data['data_seleksi'] = $this->pengambilan->select($id_p);
        $this->load->view('dsn/v_seleksimhs', $data);
    }
    else{
        $data['STATUS'] = $this->input->post('status');
        $status = $this->input->post('status');
        $id_p = $this->input->post('id_p');
        $ID_IDE = $this->input->post('id_ide');
        $this->pengambilan->update_status($id_p, $data);
        if($status == 'N')
        {
            $this->pengambilan->add_kuota($ID_IDE);
        }
        redirect('Dosen/seleksi');
    }
}

```

Gambar 4. 72 Coding seleksi mahasiswa

```

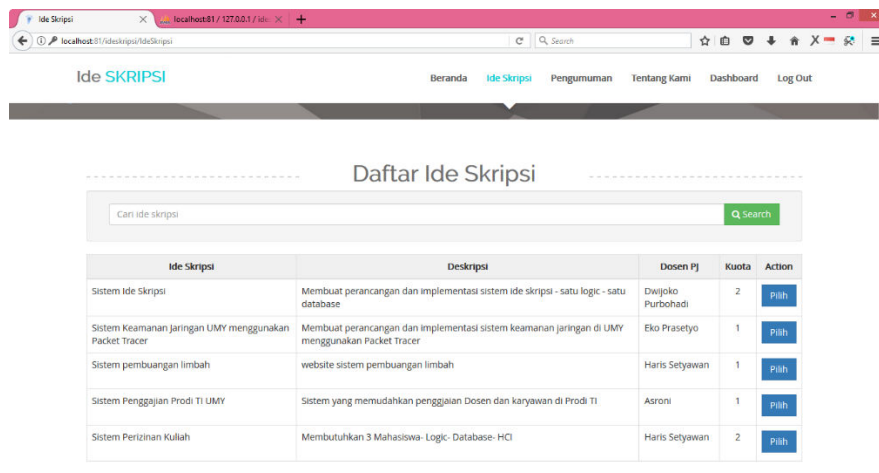
function select($id)
{
    return $this->db->get_where('pengambilanide', array('ID_PIS'=>$id))->row();
}
function update_status($id, $data)
{
    $this->db->where(array('ID_PIS' => $id, 'ID_PIS !=' => 0));
    $this->db->update('pengambilanide', $data);
}

```

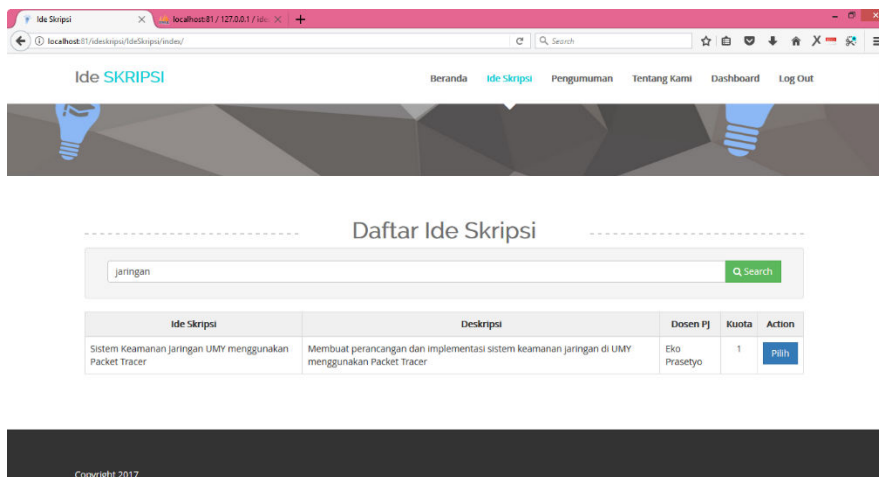
Gambar 4. 73 Fungsi *select* dan *update_status*

4.2.8 Halaman Ide Skripsi

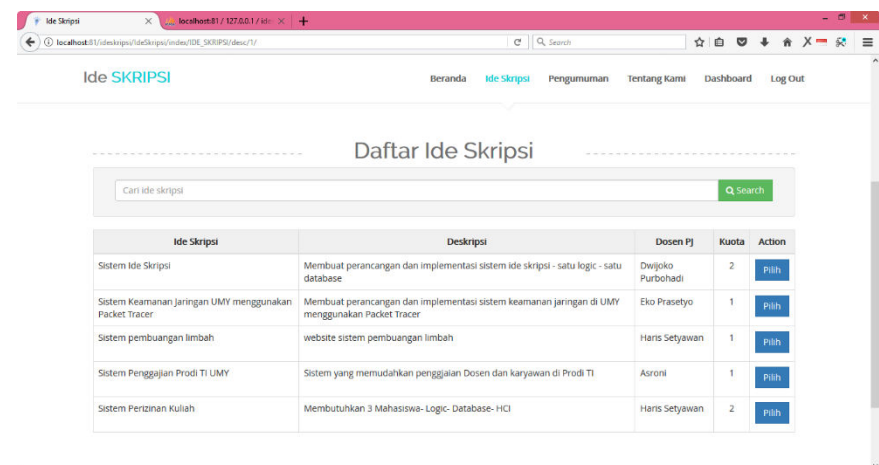
Halaman ide skripsi menampilkan daftar ide skripsi dosen yang dapat diambil oleh mahasiswa. Pada halaman ini, dapat dilakukan pencarian ide skripsi dengan mengetikkan kata kunci atau mengurutkan data ide skripsi dengan mengklik *header cell* pada tabel. Mahasiswa dapat mengambil ide skripsi dengan mengklik pilih pada ide skripsi yang ingin diambil. Kemudian halaman baru tertampil untuk menanyakan kembali apakah mahasiswa yakin mengambil ide skripsi tersebut. Jika iya, mahasiswa mengklik *submit*. Lalu, muncul notifikasi bahwa mahasiswa telah mengambil ide skripsi tersebut dan dipersilakan menunggu pengumuman. Tampilan dapat dilihat pada Gambar 4.74, Gambar 4.75, Gambar 4.76, Gambar 4.77, Gambar 4.78.

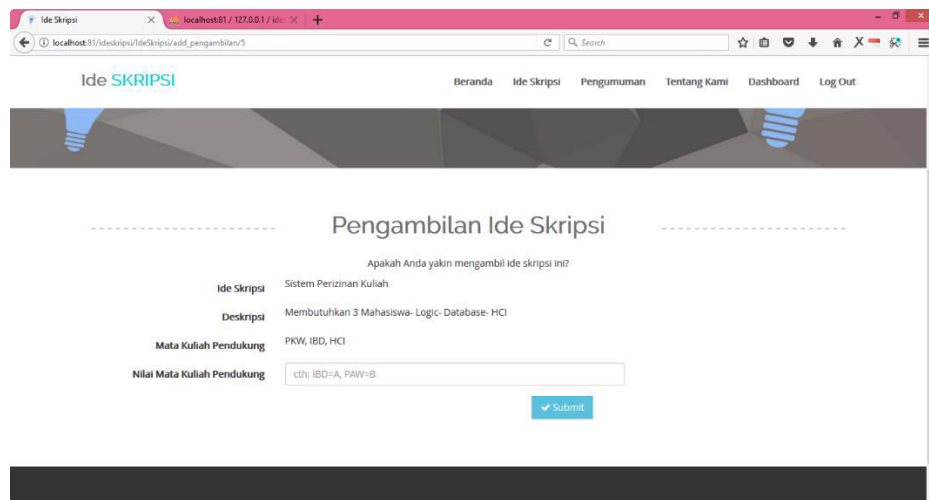


Gambar 4. 74 Tampilan daftar ide skripsi

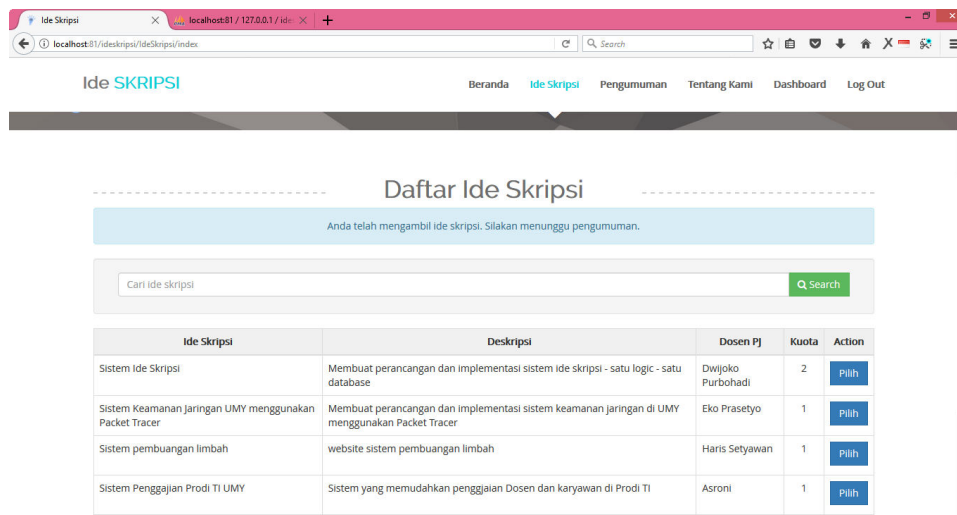


Gambar 4. 75 Tampilan pencarian ide skripsi

Gambar 4. 76 Tampilan *sorting* ide skripsi berdasarkan judul



Gambar 4. 77 Tampilan pengambilan ide skripsi



Gambar 4. 78 Tampilan notifikasi pengambilan ide skripsi

Pada Gambar 4.79, dapat dilihat bahwa fungsi *index* dalam *controller* IdeSkripsi memanggil fungsi *get_ide* dan *record_count* yang ada pada *model* ide. Fungsi *get_ide* digunakan untuk mendapatkan data ide skripsi dari *database*. Sedangkan fungsi *record_count* digunakan untuk menghitung jumlah data ide skripsi. *Coding* fungsi *get_ide* dan *record_count* dalam *model* ide dapat dilihat pada Gambar 4.80.

```

function index()
{
    $data = array();
    $data['title'] = 'Ide Skripsi';
    $data['sort_cols'] = array(
        'IDE_SKRIPSI' => 'Ide Skripsi',
        'DESKRIPSI' => 'Deskripsi',
        'NAMA_DSN' => 'Dosen PJ',
        'KUOTA_AMBIL' => 'Kuota'
    );
    $config = array();
    //base_url () . 'index.php/questions/page/'. $sortfield.'/'. $order.'/',
    $search_string = $this->input->post('search');

    $config["per_page"] = 10;
    //max number of page links
    $config['num_links'] = 2;
    //use page number as parameter
    $config['use_page_numbers'] = TRUE;

    $data['search_string'] = '';
    if(!empty($search_string)) {
        $this->uri->segment(6, $this->uri->segment(5, 1));
        $data['search_string'] = $this->uri->segment(5, $search_string);
    } elseif($this->uri->segment(5) != null && !empty($this->uri->segment(5)) && $this->uri->segment(6) != null) {
        $data['search_string'] = $this->uri->segment(5);
    }
    //set default page uri
    $page_uri = 5;

    if(!empty($data['search_string']))
        $page_uri = 6;

    $config["uri_segment"] = $page_uri;

    $config["total_rows"] = $this->ide->record_count($data['search_string']);

    $data['page'] = $this->uri->segment($page_uri, 1);

    $data['sort_by'] = $this->uri->segment(3, 'IDE_SKRIPSI');
    $orderBy = $this->uri->segment(4, "desc");
    $offset = ($data['page']-1) * $config['per_page'];
    $data['total_rows'] = $config["total_rows"];
    if($orderBy == "asc") $data['sort_order'] = "desc"; else $data['sort_order'] = "asc";

    $config["base_url"] = base_url(). 'IdeSkripsi/index/'. $data['sort_by'].'/'. $orderBy.'/'. $data['search_string'];
    $config['full_tag_open'] = '<ul class="pagination">';
    $config['full_tag_close'] = '</ul>';
    $config['first_link'] = '&laquo; First';
    $config['first_tag_open'] = '<li class="prev page">';
    $config['first_tag_close'] = '</li>';

    $config['last_link'] = 'Last &raquo;';
    $config['last_tag_open'] = '<li class="next page">';
    $config['last_tag_close'] = '</li>';

    $config['next_link'] = 'Next &rarr;';
    $config['next_tag_open'] = '<li class="next page">';
    $config['next_tag_close'] = '</li>';

    $config['prev_link'] = '&larr; Previous';
    $config['prev_tag_open'] = '<li class="prev page">';
    $config['prev_tag_close'] = '</li>';

    $config['cur_tag_open'] = '<li class="active"><a href="">';
    $config['cur_tag_close'] = '</a></li>';

    $config['num_tag_open'] = '<li class="page">';
    $config['num_tag_close'] = '</li>';

    $data["data"] = $this->ide->get_ide($config["per_page"], $offset, $data['sort_by'], $data['sort_order'], $data['search_string']);

    $this->pagination->initialize($config);
    $data["links"] = $this->pagination->create_links();

    $this->load->view('v_idekripsi', $data);
}

```

Gambar 4. 79 Coding data ide skripsi

```

function get_ide($per_page, $offset, $sortfield, $orderBy, $search_string, $id=0)
{
    if(empty($id)){
        //echo $per_page.'fff'.$offset.'fff'.$sortfield.'fff'.$orderBy;
        if(!empty($search_string)) {
            $this->db->like('IDE_SKRIPSI',$search_string);
            $this->db->or_like('DESKRIPSI',$search_string);
        }
        $this->db->order_by("$sortfield", "$orderBy");
        $this->db->limit($per_page,$offset);
        $this->db->select("*");
        $this->db->from('ideskripsi');
        $this->db->join('dosen', 'dosen.NIDN_DSN = ideskripsi.NIDN_DSN');
        $this->db->where('KUOTA>', '0');
        $query = $this->db->get();
        if ($query->num_rows() > 0) {
            foreach ($query->result() as $row) {
                $data[] = $row;
            }
            return $data;
        }
        return false;
    } else {
        $query = $this->db->get_where('ideskripsi', array('ID_IDE' => $id));
        return $query->row_array();
    }
}

public function record_count($search_string) {
    if(!empty($search_string)) {
        $this->db->like('ideskripsi.IDE_SKRIPSI',$search_string);
        $this->db->or_like('ideskripsi.DESKRIPSI',$search_string);
    }
    return $this->db->count_all_results("ideskripsi");
}

```

Gambar 4. 80 Coding model data ide skripsi

Pada Gambar 4.81, dapat dilihat bahwa fungsi `add_pengambilan` dalam *controller* `IdeSkripsi` memanggil `add_pengambilan`, `get_mhs`, `get_mhy`, dan `update_kuota` yang ada pada *model* pengambilan. Fungsi `add_pengambilan` digunakan untuk menambah data pengambil ide skripsi ke *database*. Fungsi `get_mhs` dan `get_mhy` digunakan untuk mendapatkan data pengambil ide skripsi dengan status NULL atau Y. Fungsi `update_kuota` digunakan untuk memperbarui kuota setelah ide skripsi diambil oleh mahasiswa. Coding `add_pengambilan`, `get_mhs`, `get_mhy`, dan `update_kuota` dalam *model* pengambilan dapat dilihat pada Gambar 4.82.

```

public function add_pengambilan($id)
{
    if($this->session->userdata('tipeuser')== '2') {

        $data["mhs"] = $this->pengambilan->get_mhs($this->session->userdata('id_user'));
        $data["mhy"] = $this->pengambilan->get_mhy($this->session->userdata('id_user'));

        $this->form_validation->set_rules('nilai_matkul', 'NILAI MATKUL', 'required');

        if($this->form_validation->run() == FALSE) {
            $data['data_ide'] = $this->ide->select($id);
            $this->load->view('v_pengambilan', $data);
        }
        else{

            $ID_IDE = $this->input->post('id_ide');
            $NIM_MHS = $this->input->post('nim');
            $NILAI_MATKUL = $this->input->post('nilai_matkul');
            $data = array( 'ID_IDE'=>$ID_IDE, 'NIM_MHS'=>$NIM_MHS, 'NILAI_MATKUL'=> $NILAI_MATKUL);

            $this->pengambilan->add_pengambilan($data);
            $this->pengambilan->update_kuota($ID_IDE);

            $this->session->set_flashdata('message', 'Anda telah mengambil ide skripsi. Silakan menunggu pengumuman.');
```

Gambar 4. 81 Coding tambah pengambilan

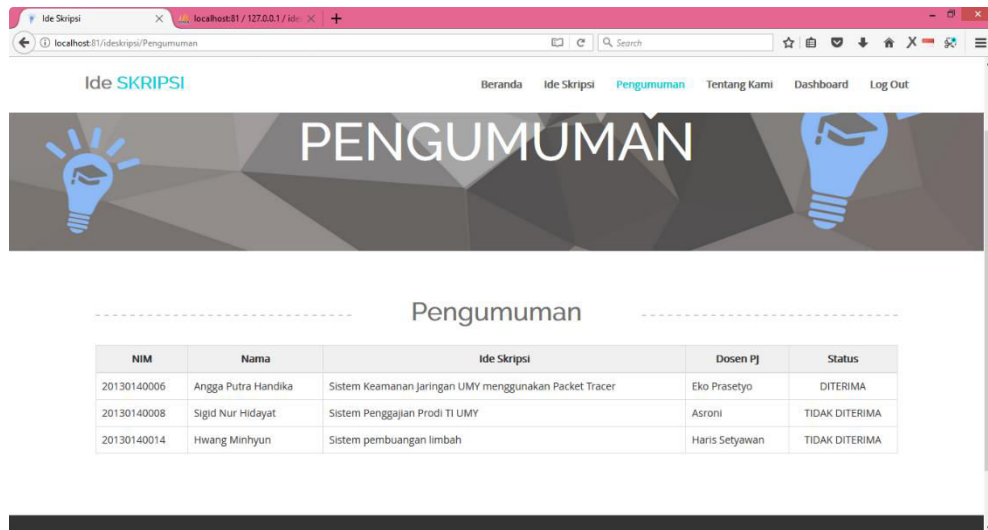
```

function add_pengambilan($data)
{
    $this->db->insert('pengambilanide',$data);
}
function get_mhs($id_user)
{
    return $this->db->get_where('pengambilanide', array('NIM_MHS'=>$id_user, 'STATUS'=> NULL))->row();
}
function get_mhy($id_user)
{
    return $this->db->get_where('pengambilanide', array('NIM_MHS'=>$id_user, 'STATUS'=> 'Y'))->row();
}
function update_kuota($id)
{
    $this->db->set('KUOTA', 'KUOTA-1', false);
    $this->db->where(array('ID_IDE' => $id, 'ID_IDE !=' => 0));
    $this->db->update('ideskripsi');
```

Gambar 4. 82 Coding model pengambilan

4.2.9 Halaman Pengumuman

Halaman pengumuman menampilkan hasil seleksi mahasiswa pengambil ide skripsi. Pada halaman ini mahasiswa dapat melihat apakah mereka diterima atau tidak untuk mengambil ide skripsi dosen. Tampilan halaman pengumuman dapat dilihat pada Gambar 4.83.



NIM	Nama	Ide Skripsi	Dosen PJ	Status
20130140006	Angga Putra Handika	Sistem Keamanan Jaringan UMY menggunakan Packet Tracer	Eko Prasetyo	DITERIMA
20130140008	Sigid Nur Hidayat	Sistem Penggajian Prodi TI UMY	Asroni	TIDAK DITERIMA
20130140014	Hwang Minhyun	Sistem pembuangan limbah	Haris Setyawan	TIDAK DITERIMA

Gambar 4. 83 Tampilan pengumuman

Pada Gambar 4.84, dapat dilihat bahwa fungsi *index* dalam *controller* Pengumuman memanggil fungsi *get_pengambilan* yang ada pada *model* pengambilan. Fungsi *get_pengumuman* digunakan untuk mendapatkan data hasil seleksi mahasiswa pengambil ide skripsi dari *database*. *Coding* fungsi *get_pengumuman* dapat dilihat pada Gambar 4.85.

```

public function index()
{
    $data = array();
    $data['title'] = 'Pengumuman';
    $data['sort_cols'] = array(
        'NIM_MHS' => 'NIM',
        'NAMA_MHS' => 'Nama',
        'IDE_SKRIPSI' => 'Ide Skripsi',
        'NAMA_DSN' => 'Dosen PJ',
        'STATUS' => 'Status'
    );
    $config = array();

    $data['sort_by'] = $this->uri->segment(3, 'NIM_MHS');
    $orderBy = $this->uri->segment(4, "desc");

    if($orderBy == "asc") $data['sort_order'] = "desc"; else $data['sort_order'] = "asc";

    $data["data"] = $this->pengumuman->get_pengumuman($data['sort_by'], $data['sort_order']);

    $this->load->view('v_pengumuman', $data);
}

```

Gambar 4. 84 Coding controller pengumuman

```

function get_pengumuman($sortfield, $orderby, $id=0)
{
    if(empty($id)){
        $this->db->order_by("$sortfield", "$orderby");
        $this->db->select("pengambilanide.ID_PIS, pengambilanide.STATUS, pengambilanide.ID_IDE, ideskripsi.IDE_SKRIPSI, pengambilanide.NIM_MHS");
        $this->db->from('pengambilanide');
        $this->db->join('ideskripsi', 'ideskripsi.ID_IDE = pengambilanide.ID_IDE');
        $this->db->join('mahasiswa', 'mahasiswa.NIM_MHS = pengambilanide.NIM_MHS');
        $this->db->join('dosen', 'ideskripsi.NIDN_DSN = dosen.NIDN_DSN');
        $this->db->where("(STATUS = 'Y' OR STATUS = 'N')");

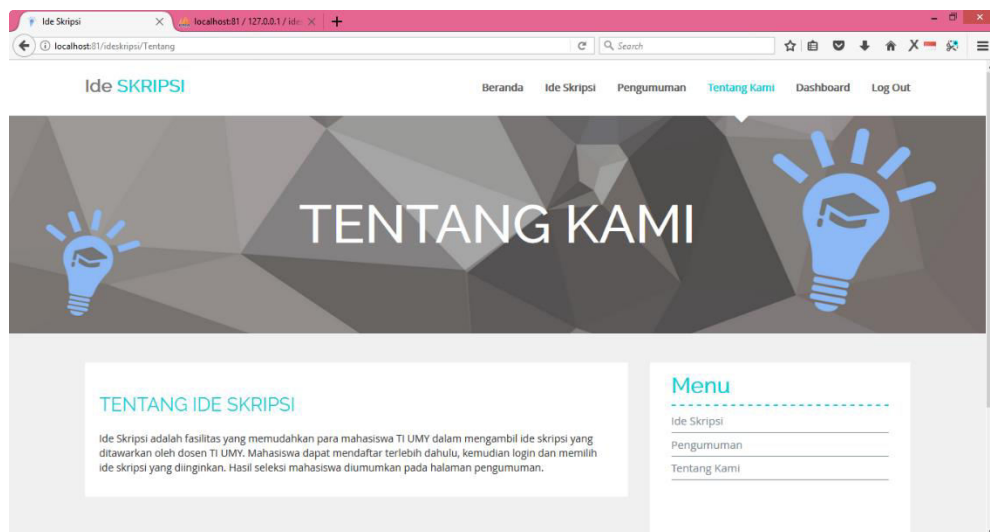
        $query = $this->db->get();
        if ($query->num_rows() > 0) {
            foreach ($query->result() as $row) {
                $data[] = $row;
            }
            return $data;
        }
        return false;
    } else {
        $query = $this->db->get_where('pengambilanide', array('ID_PIS' => $id));
        return $query->row_array();
    }
}

```

Gambar 4. 85 Coding model pengumuman

4.2.10 Halaman Tentang Kami

Halaman tentang kami adalah halaman yang menampilkan penjelasan singkat mengenai *website* ide skripsi. Tampilan halaman tentang kami dapat dilihat pada Gambar 4.86.



Gambar 4. 86 Tampilan halaman tentang kami

Gambar 4.87 memperlihatkan *coding controller* Tentang yang memanggil tampilan tentang dalam folder *views*. *Coding* tampilan tentang dapat dilihat pada Gambar 4.88.

```

class Tentang extends CI_Controller {
    public function index()
    {
        $this->load->view('tentang');
    }
}

```

Gambar 4. 87 Coding controller tentang

```

<?php
$currentPage= 'tentang';
include('/layout/header.php');
?>

<!-- start banner area -->
<section id="imgbanner">
    <h2>Tentang Kami</h2>
</section>
<!-- End banner area -->

<!-- start image editing -->
<section id="blogarchive">
    <div class="container">
        <div class="row">
            <div class="col-lg-8 col-md-8 col-sm-12">
                <div class="blogArchive_area">
                    <!-- start page left side -->
                    <div class="single_archiveblog">
                        <div class="archiveblog_right page_left">
                            <h2>Tentang Ide Skripsi</h2>
                            <p>Ide Skripsi adalah fasilitas yang memudahkan para mahasiswa TI UMY dalam mengambil ide skripsi yang ditawarkan oleh dosen TI Mahasiswa dapat mendaftar terlebih dahulu, kemudian login dan memilih ide skripsi yang diinginkan. Hasil seleksi mahasiswa di
                            </p>
                        </div>
                    </div>
                    <!-- End page left side -->
                </div>
            </div>
            <div class="col-lg-4 col-md-4 col-sm-12">
                <div class="blog_sidebar">
                    <!-- Start single side bar -->
                    <div class="single_sidebar">
                        <h2>Menu</h2>
                        <ul class="catg_nav">
                            <li><a href="<?php echo base_url(). 'IdeSkripsi';?>">Ide Skripsi</a></li>
                            <li><a href="<?php echo base_url(). 'Pengumuman';?>">Pengumuman</a></li>
                            <li><a href="<?php echo base_url(). 'Tentang';?>">Tentang Kami</a></li>
                        </ul>
                    </div>
                </div>
            </div>
        </div>
    </div>

```

Gambar 4. 88 Tampilan halaman tentang kami


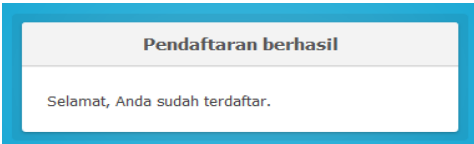

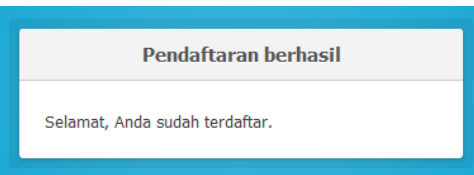

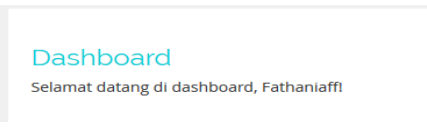


4.3 Pengujian Sistem


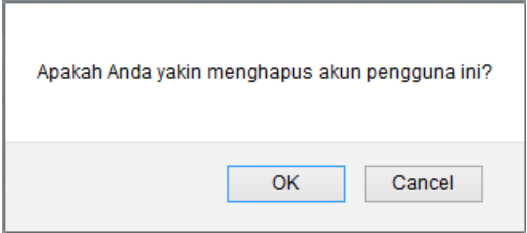




Pengujian sistem dilakukan untuk mengetahui apakah sistem yang telah dibuat sudah sesuai dengan yang diharapkan. Pengujian juga dilakukan untuk menghindari kesalahan-kesalahan yang mungkin terjadi ketika digunakan oleh pengguna.


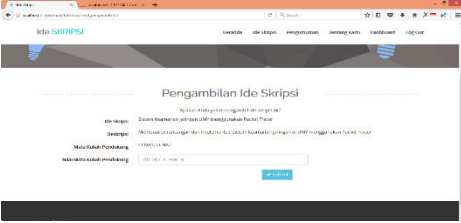

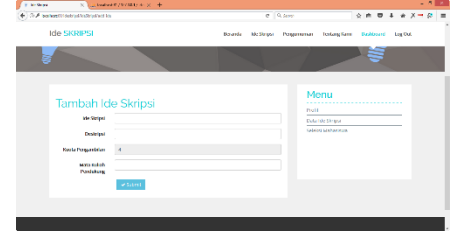

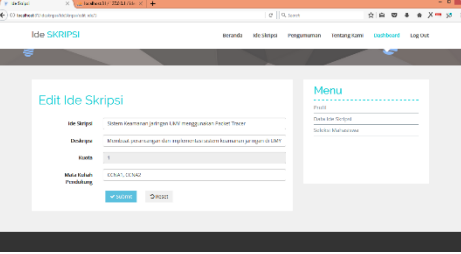
4.3.1 Pengujian User Interface


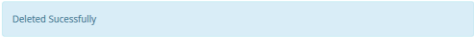

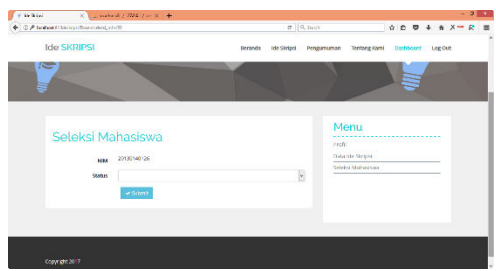
Pengujian *user interface* ditujukan untuk mengetahui fungsionalitas tiap elemen *interface* yang terdapat dalam halaman sistem. Elemen yang diujikan adalah elemen *button* pada setiap halaman *website*. Hasil pengujian dapat dilihat pada Tabel 4.1.

Tabel 4. 1 Hasil pengujian *user interface*

No.	Kasus yang diuji	Test Case	Hasil yang diharapkan	Hasil yang didapatkan	Status
1.	Tombol daftar pada halaman registrasi mahasiswa		Sistem menampilkan halaman berhasil mendaftarkan akun.		Berhasil
2.	Tombol daftar pada halaman registrasi dosen		Sistem menampilkan halaman berhasil mendaftarkan akun.		Berhasil
3.	Tombol <i>login</i>		Sistem dapat menampilkan halaman <i>dashboard</i> .		Berhasil
4.	Tombol edit pada manajemen akun dalam <i>dashboard</i> administrator.		Sistem dapat menampilkan halaman edit data.		Berhasil

No.	Kasus yang diuji	Test Case	Hasil yang diharapkan	Hasil yang didapatkan	Status
5.	Tombol hapus pada manajemen akun dalam <i>dashboard</i> administrator.		Sistem dapat menampilkan kotak dialog penghapusan.		Berhasil
6.	Tombol aktivasi pada manajemen akun dalam <i>dashboard</i> administrator		Sistem dapat menampilkan notifikasi aktivasi akun.		Berhasil
7.	Tombol <i>reset</i> pada halaman ubah data		Sistem dapat mengembalikan nilai pada input.		Berhasil

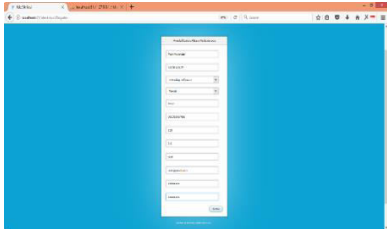
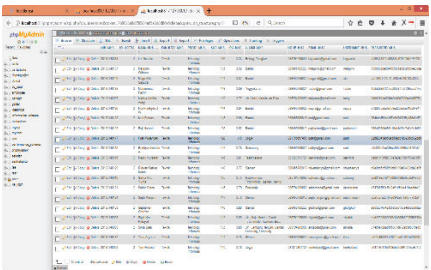
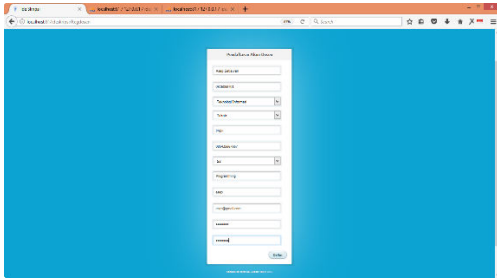
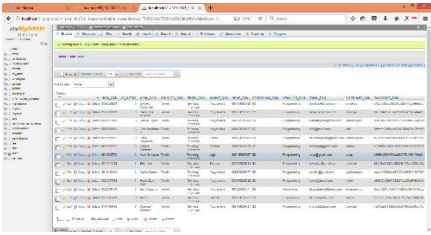
No.	Kasus yang diuji	Test Case	Hasil yang diharapkan	Hasil yang didapatkan	Status
8.	Tombol pilih pada halaman ide skripsi		Sistem menampilkan halaman pengambilan ide skripsi.		Berhasil
9.	Tombol tambah ide skripsi pada halaman data ide skripsi dalam <i>dashboard</i> dosen		Sistem menampilkan halaman tambah ide skripsi		Berhasil
10.	Tombol edit pada halaman data ide skripsi dalam <i>dashboard</i> dosen		Sistem dapat menampilkan halaman edit ide skripsi		Berhasil

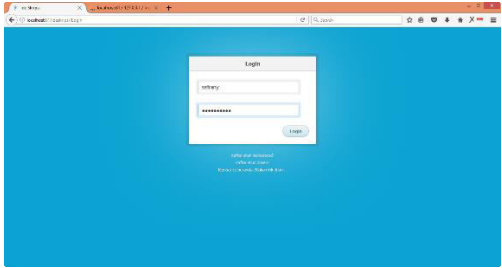
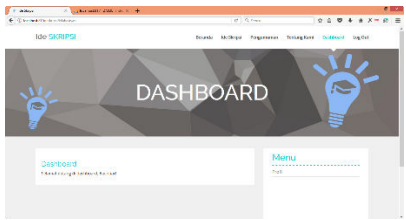
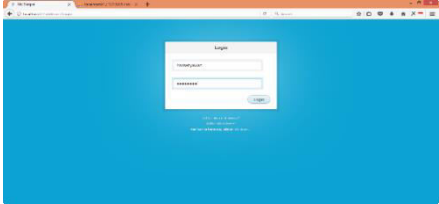
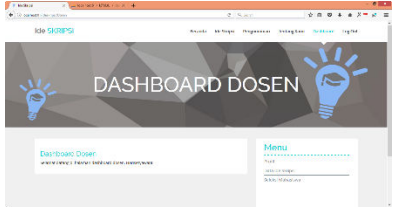
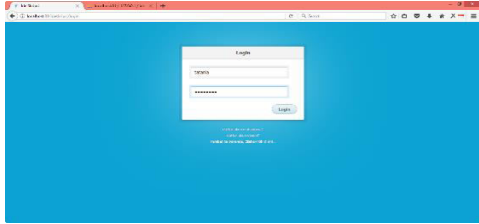
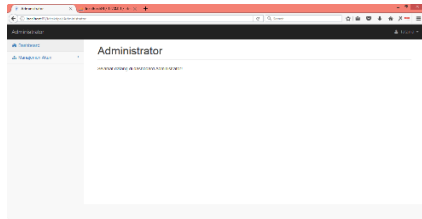
No.	Kasus yang diuji	Test Case	Hasil yang diharapkan	Hasil yang didapatkan	Status
11.	Tombol hapus pada halaman data ide skripsi dalam <i>dashboard</i> dosen		Sistem dapat menampilkan notifikasi penghapusan.		Berhasil
12.	Tombol pilih pada halaman seleksi mahasiswa skripsi dalam <i>dashboard</i> dosen		Sistem menampilkan halaman seleksi mahasiswa		Berhasil

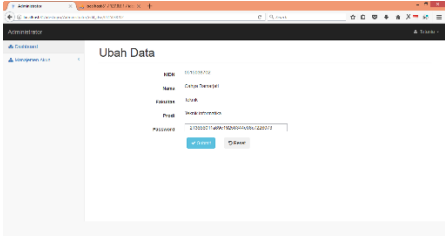
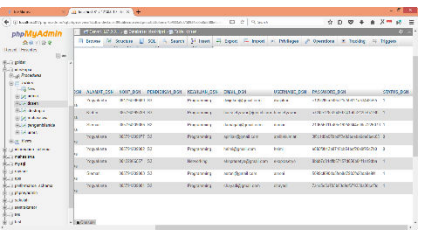
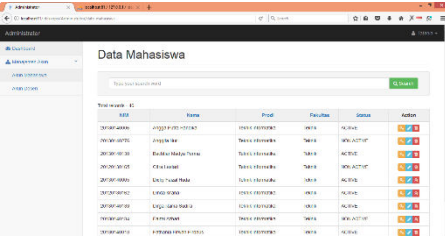
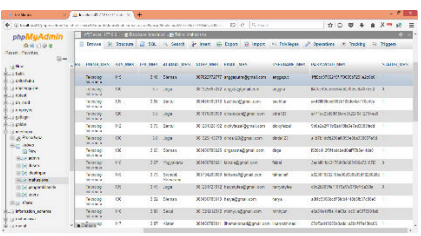
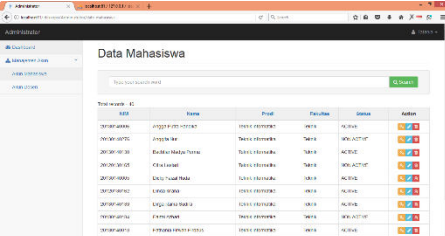
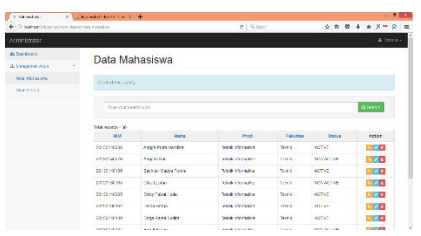
4.3.2 Pengujian Fungsi Sistem

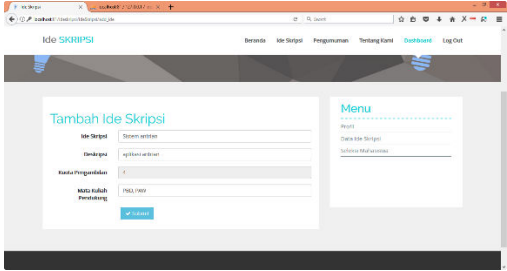
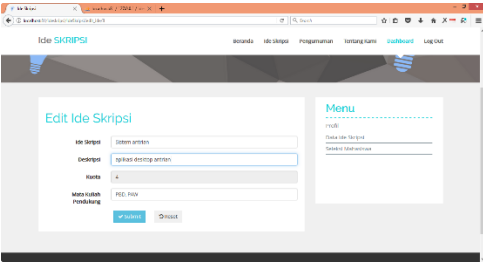
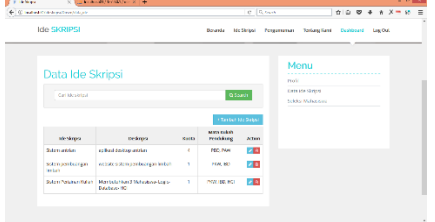
Pengujian fungsi sistem ditujukan untuk mengetahui apakah fungsi-fungsi yang ada dalam sistem sudah bekerja seperti yang seharusnya. Hasil pengujian dapat dilihat pada tabel 4.2.

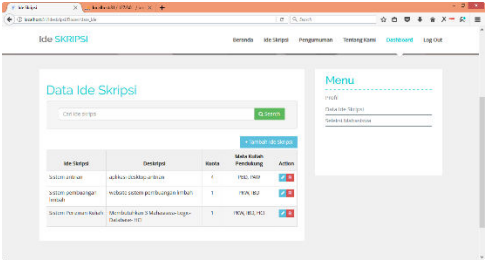
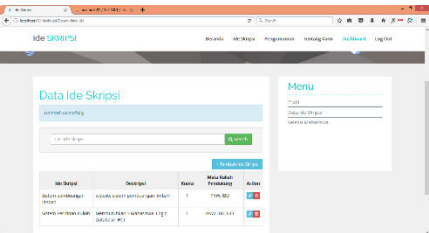
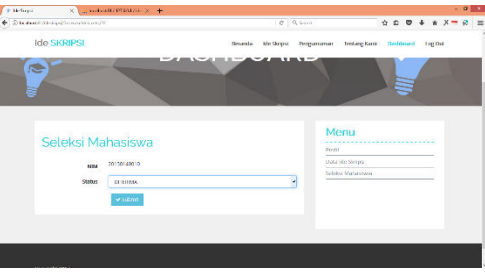
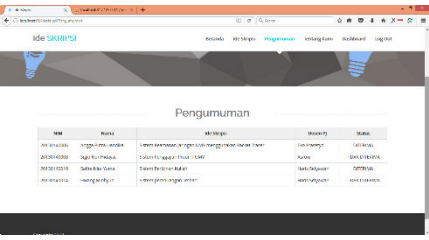
Tabel 4. 2 Hasil pengujian fungsi sistem

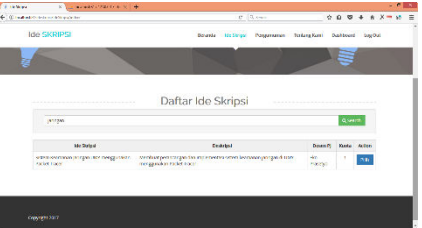
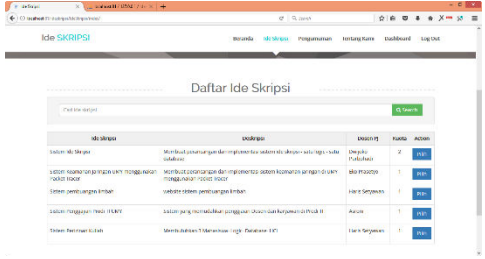
No.	Kasus yang diuji	Test Case	Hasil yang diharapkan	Hasil yang didapatkan	Status
1.	Pengujian fungsi register mahasiswa		Sistem dapat menambah mahasiswa dalam basis data.		Berhasil
2.	Pengujian fungsi register dosen		Sistem dapat menambah dosen dalam basis data.		Berhasil

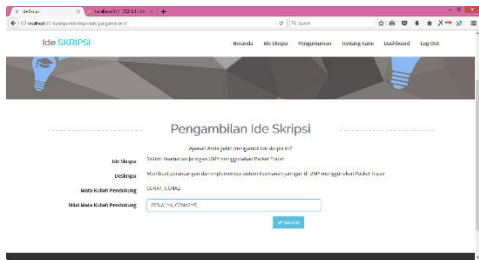
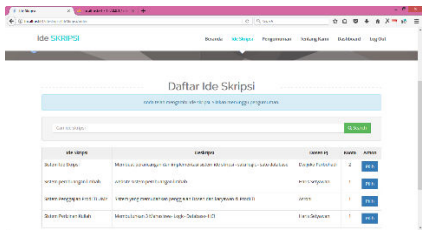
No.	Kasus yang diuji	Test Case	Hasil yang diharapkan	Hasil yang didapatkan	Status
3.	Mahasiswa melakukan <i>login</i>		Mahasiswa dapat masuk ke dalam dashboard.		Berhasil
4.	Dosen melakukan <i>login</i>		Dosen dapat masuk ke dalam dashboard dosen.		Berhasil
5.	Administrator melakukan <i>login</i>		Administrator dapat masuk ke dalam dashboard administrator.		Berhasil

No.	Kasus yang diuji	Test Case	Hasil yang diharapkan	Hasil yang didapatkan	Status
6.	Fungsi ubah <i>password</i> dalam manajemen akun		Sistem dapat mengubah <i>password</i> akun dan menyimpannya dalam enkripsi md5.		Berhasil
7.	Fungsi aktivasi akun dalam manajemen akun		Sistem dapat mengubah status akun pengguna menjadi aktif.		Berhasil
8.	Fungsi hapus dalam manajemen akun		Sistem dapat menghapus data akun.		Berhasil

No.	Kasus yang diuji	Test Case	Hasil yang diharapkan	Hasil yang didapatkan	Status
12.	Fungsi tambah ide skripsi pada menu data ide skripsi dalam <i>dashboard</i> dosen		Sistem dapat menambah data ide skripsi dosen.		Berhasil
13.	Fungsi ubah ide skripsi pada menu data ide skripsi dalam <i>dashboard</i> dosen		Sistem dapat mengubah data ide skripsi dosen.		Berhasil

No.	Kasus yang diuji	Test Case	Hasil yang diharapkan	Hasil yang didapatkan	Status
14.	Fungsi hapus ide skripsi pada menu data ide skripsi dalam <i>dashboard</i> dosen		Sistem dapat menghapus data ide skripsi dosen.		Berhasil
15.	Fitur seleksi mahasiswa pada menu seleksi mahasiswa dalam <i>dashboard</i> dosen		Sistem dapat menampilkan hasil seleksi pada halaman pengumuman.		Berhasil

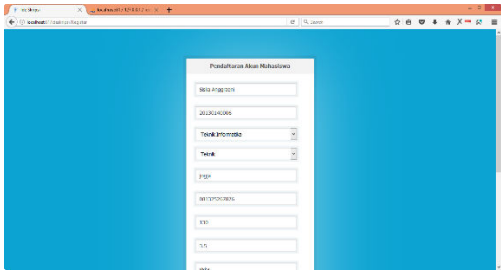
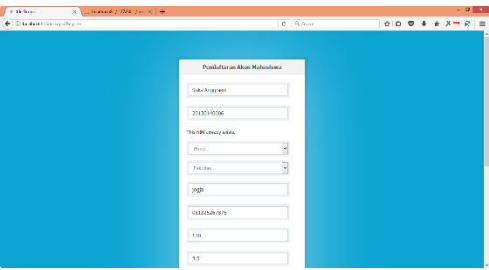
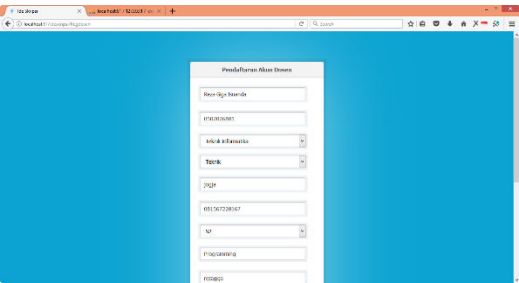
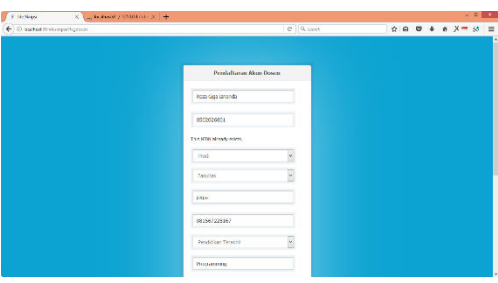
No.	Kasus yang diuji	Test Case	Hasil yang diharapkan	Hasil yang didapatkan	Status																									
16.	Fitur pencarian pada halaman ide skripsi	 <table border="1" data-bbox="577 639 972 756"> <thead> <tr> <th>Uraian</th> <th>Detail</th> <th>Detail</th> <th>Kasus</th> <th>Aksi</th> </tr> </thead> <tbody> <tr> <td>Sistem: Ide Skripsi</td> <td>Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan</td> <td>Detail: Pustaka</td> <td>1</td> <td>Uji</td> </tr> <tr> <td>Sistem: Sistem pencarian yang relevan</td> <td>Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan</td> <td>Detail: Pustaka</td> <td>1</td> <td>Uji</td> </tr> <tr> <td>Sistem: Pencarian yang relevan</td> <td>Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan</td> <td>Detail: Pustaka</td> <td>1</td> <td>Uji</td> </tr> <tr> <td>Sistem: Pencarian yang relevan</td> <td>Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan</td> <td>Detail: Pustaka</td> <td>1</td> <td>Uji</td> </tr> </tbody> </table>	Uraian	Detail	Detail	Kasus	Aksi	Sistem: Ide Skripsi	Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan	Detail: Pustaka	1	Uji	Sistem: Sistem pencarian yang relevan	Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan	Detail: Pustaka	1	Uji	Sistem: Pencarian yang relevan	Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan	Detail: Pustaka	1	Uji	Sistem: Pencarian yang relevan	Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan	Detail: Pustaka	1	Uji	Sistem dapat menampilkan hasil pencarian sesuai kata kunci pencarian.		Berhasil
Uraian	Detail	Detail	Kasus	Aksi																										
Sistem: Ide Skripsi	Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan	Detail: Pustaka	1	Uji																										
Sistem: Sistem pencarian yang relevan	Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan	Detail: Pustaka	1	Uji																										
Sistem: Pencarian yang relevan	Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan	Detail: Pustaka	1	Uji																										
Sistem: Pencarian yang relevan	Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan	Detail: Pustaka	1	Uji																										
17.	Fitur <i>sorting</i> pada halaman ide skripsi	 <table border="1" data-bbox="577 1086 972 1203"> <thead> <tr> <th>Uraian</th> <th>Detail</th> <th>Detail</th> <th>Kasus</th> <th>Aksi</th> </tr> </thead> <tbody> <tr> <td>Sistem: Ide Skripsi</td> <td>Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan</td> <td>Detail: Pustaka</td> <td>1</td> <td>Uji</td> </tr> <tr> <td>Sistem: Sistem pencarian yang relevan</td> <td>Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan</td> <td>Detail: Pustaka</td> <td>1</td> <td>Uji</td> </tr> <tr> <td>Sistem: Pencarian yang relevan</td> <td>Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan</td> <td>Detail: Pustaka</td> <td>1</td> <td>Uji</td> </tr> <tr> <td>Sistem: Pencarian yang relevan</td> <td>Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan</td> <td>Detail: Pustaka</td> <td>1</td> <td>Uji</td> </tr> </tbody> </table>	Uraian	Detail	Detail	Kasus	Aksi	Sistem: Ide Skripsi	Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan	Detail: Pustaka	1	Uji	Sistem: Sistem pencarian yang relevan	Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan	Detail: Pustaka	1	Uji	Sistem: Pencarian yang relevan	Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan	Detail: Pustaka	1	Uji	Sistem: Pencarian yang relevan	Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan	Detail: Pustaka	1	Uji	Sistem dapat menampilkan data urut dengan pilihan <i>sorting</i> .		Berhasil
Uraian	Detail	Detail	Kasus	Aksi																										
Sistem: Ide Skripsi	Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan	Detail: Pustaka	1	Uji																										
Sistem: Sistem pencarian yang relevan	Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan	Detail: Pustaka	1	Uji																										
Sistem: Pencarian yang relevan	Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan	Detail: Pustaka	1	Uji																										
Sistem: Pencarian yang relevan	Mencari ide skripsi yang relevan dengan kata kunci yang dimasukkan	Detail: Pustaka	1	Uji																										

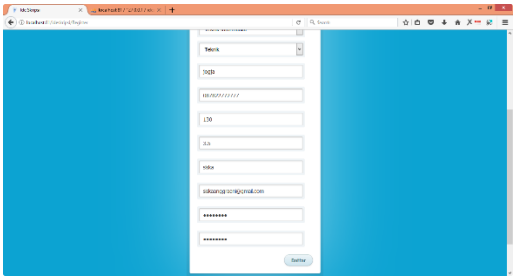
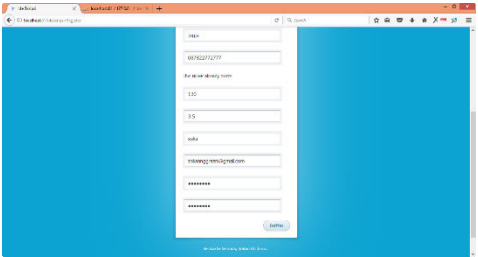
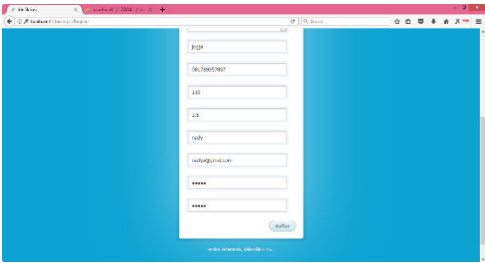
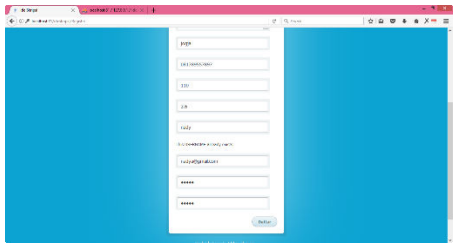
No.	Kasus yang diuji	Test Case	Hasil yang diharapkan	Hasil yang didapatkan	Status
18.	Fitur ambil ide skripsi oleh mahasiswa		Sistem dapat menerima pengambilan ide skripsi oleh mahasiswa dan menampilkan notifikasi. Kuota yang tersedia pun berkurang setelah ide skripsi diambil oleh mahasiswa.		Berhasil

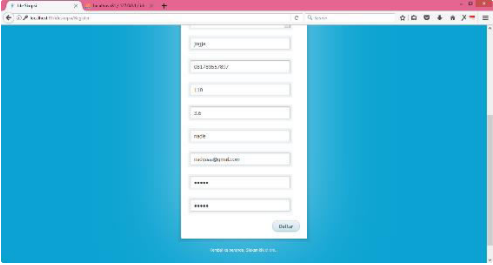
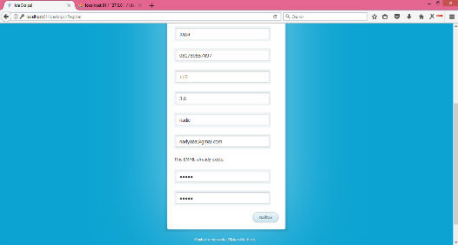
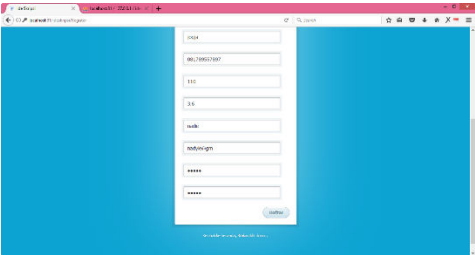
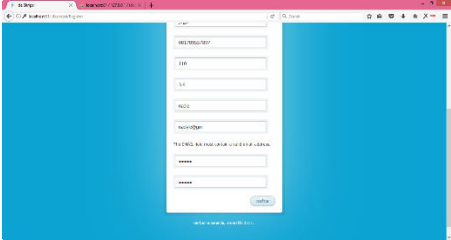
4.3.3 Pengujian Validasi

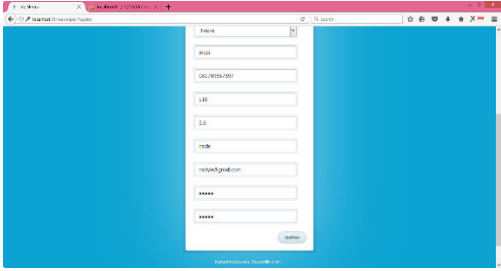
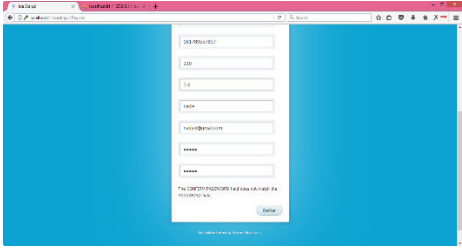
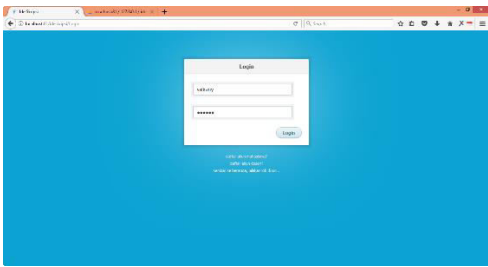
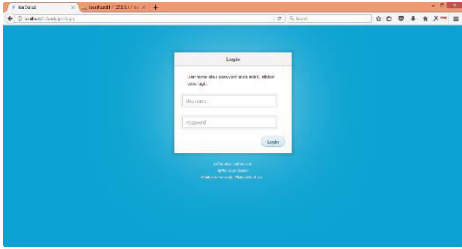
Pengujian validasi bertujuan untuk mengetahui apakah validasi-validasi yang ada dalam sistem sudah berjalan dengan baik. Hasil pengujian validasi dapat dilihat pada tabel 4.3.

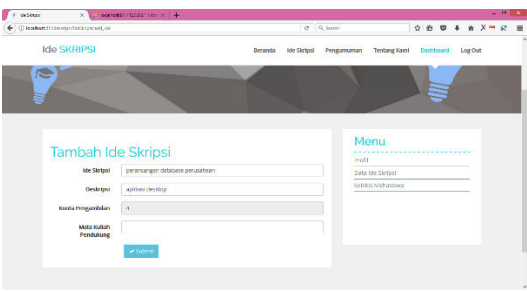
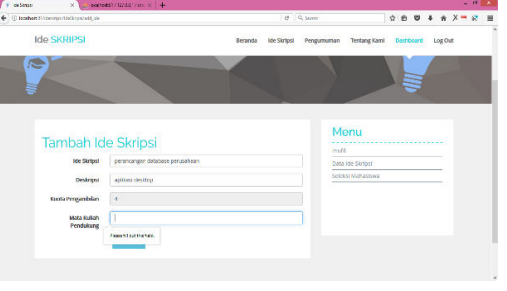
Tabel 4. 3 Hasil pengujian validasi

No.	Kasus yang diuji	Test Case	Hasil yang diharapkan	Hasil yang didapatkan	Status
1.	Validasi jika NIM sudah terdaftar		Sistem dapat menampilkan peringatan bahwa NIM sudah ada dalam sistem.		Berhasil
2.	Validasi jika NIDN sudah terdaftar		Sistem dapat menampilkan peringatan bahwa NIDN sudah ada dalam sistem.		Berhasil

No.	Kasus yang diuji	Test Case	Hasil yang diharapkan	Hasil yang didapatkan	Status
3.	Validasi jika nomor hp sudah terdaftar		Sistem dapat menampilkan peringatan bahwa nomor hp sudah ada dalam sistem.		Berhasil
4.	Validasi jika <i>username</i> sudah terdaftar		Sistem dapat menampilkan peringatan bahwa <i>username</i> sudah ada dalam sistem.		Berhasil

No.	Kasus yang diuji	Test Case	Hasil yang diharapkan	Hasil yang didapatkan	Status
5.	Validasi jika <i>email</i> sudah terdaftar		Sistem dapat menampilkan peringatan bahwa <i>email</i> sudah ada dalam sistem.		Berhasil
6.	Validasi <i>email</i> jika tidak valid		Sistem dapat menampilkan peringatan bahwa <i>email</i> tidak valid.		Berhasil

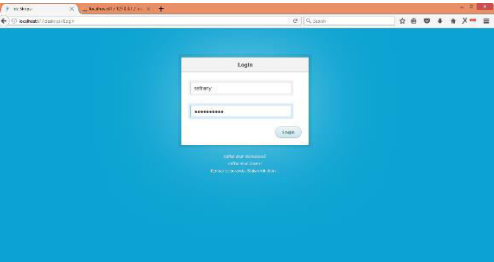
No.	Kasus yang diuji	Test Case	Hasil yang diharapkan	Hasil yang didapatkan	Status
7.	Validasi <i>confirm password</i> jika tidak sama dengan <i>password</i>		Sistem dapat menampilkan peringatan bahwa <i>confirm password</i> tidak sama dengan <i>password</i> .		Berhasil
8.	Validasi login jika <i>username</i> atau <i>password</i> salah		Sistem dapat menampilkan peringatan bahwa <i>username</i> atau <i>password</i> salah.		Berhasil

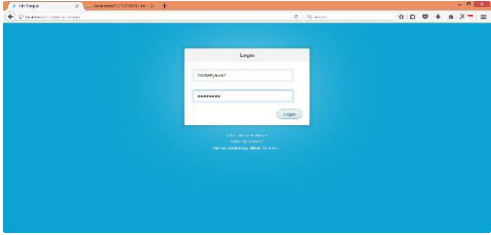
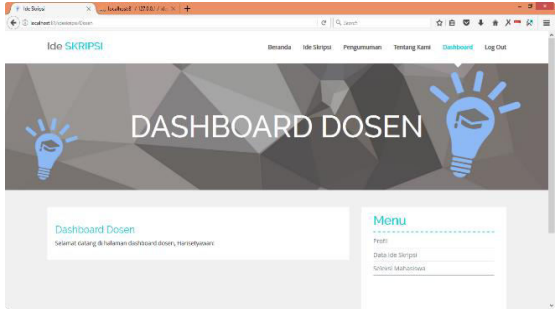
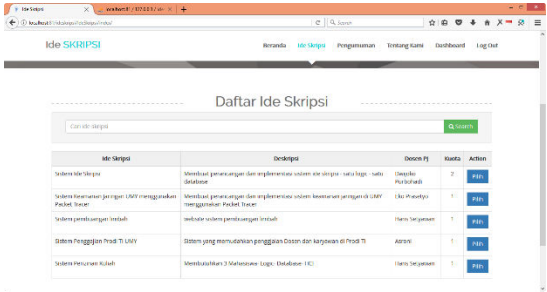
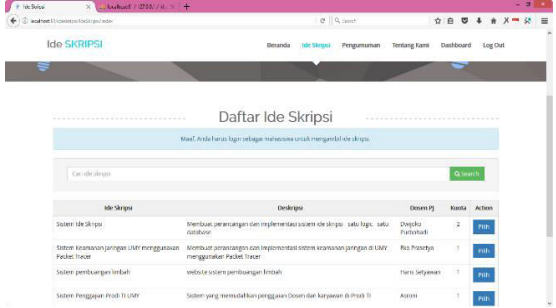
No.	Kasus yang diuji	Test Case	Hasil yang diharapkan	Hasil yang didapatkan	Status
9.	Validasi jika input ide skripsi masih kosong		Sistem dapat menampilkan peringatan bahwa input harus diisi.		Berhasil

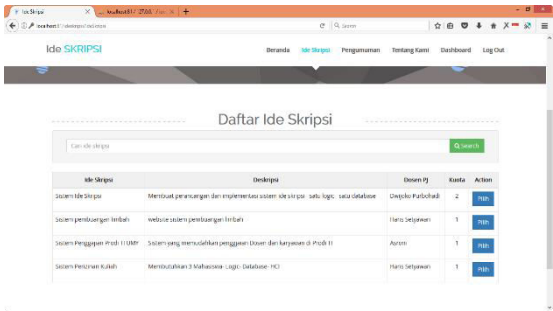
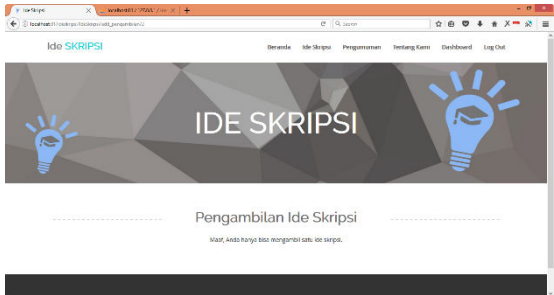
4.3.4 Pengujian Keamanan Sistem

Pengujian keamanan sistem ditujukan untuk mengetahui apakah keamanan yang diterapkan dalam sistem sudah berjalan dengan baik. Hasil pengujian keamanan sistem dapat dilihat pada Tabel 4.4.

Tabel 4. 4 Hasil pengujian keamanan sistem

No.	Kasus yang diuji	Test Case	Hasil yang diharapkan	Hasil yang didapatkan	Status
1.	Pengguna dengan level administrator		Pengguna dengan level administrator hanya dapat masuk ke <i>dashboard</i> administrator.		Berhasil
2.	Pengguna dengan level mahasiswa		<i>Dashboard</i> mahasiswa hanya bisa diakses oleh pengguna dengan level mahasiswa.		Berhasil

No.	Kasus yang diuji	Test Case	Hasil yang diharapkan	Hasil yang didapatkan	Status
3.	Pengguna dengan level dosen		<p><i>Dashboard</i></p> <p>dosen hanya bisa diakses oleh pengguna dengan level dosen.</p>		Berhasil
4.	Pengguna yang tidak <i>login</i> atau level dosen mengambil ide skripsi.		<p>Muncul notifikasi bahwa pengguna harus login sebagai mahasiswa untuk dapat mengambil ide skripsi.</p>		Berhasil

No.	Kasus yang diuji	Test Case	Hasil yang diharapkan	Hasil yang didapatkan	Status																									
5.	Mahasiswa yang sudah diterima atau sudah mengambil ide skripsi mengambil ide skripsi untuk kedua kalinya	 <table border="1" data-bbox="562 703 1010 807"> <thead> <tr> <th>Ide Skripsi</th> <th>Deskripsi</th> <th>Dosen PJ</th> <th>Kuantitas</th> <th>Action</th> </tr> </thead> <tbody> <tr> <td>Sistem Ide Skripsi</td> <td>Membuat perencanaan dan implementasi sistem ide skripsi satu login satu database</td> <td>Chapko Pabichah</td> <td>2</td> <td>Pin</td> </tr> <tr> <td>Sistem perubahan bentuk</td> <td>Website sistem perubahan bentuk</td> <td>Hario Setjawan</td> <td>1</td> <td>Pin</td> </tr> <tr> <td>Sistem Penggajian Prodi IT</td> <td>Sistem yang memudahkan penggajian dosen dan karyawan di Prodi IT</td> <td>Azzam</td> <td>1</td> <td>Pin</td> </tr> <tr> <td>Sistem Pencarian Kuliab</td> <td>Membuatkan 3 Mahasiswa Login Database HCI</td> <td>Hario Setjawan</td> <td>1</td> <td>Pin</td> </tr> </tbody> </table>	Ide Skripsi	Deskripsi	Dosen PJ	Kuantitas	Action	Sistem Ide Skripsi	Membuat perencanaan dan implementasi sistem ide skripsi satu login satu database	Chapko Pabichah	2	Pin	Sistem perubahan bentuk	Website sistem perubahan bentuk	Hario Setjawan	1	Pin	Sistem Penggajian Prodi IT	Sistem yang memudahkan penggajian dosen dan karyawan di Prodi IT	Azzam	1	Pin	Sistem Pencarian Kuliab	Membuatkan 3 Mahasiswa Login Database HCI	Hario Setjawan	1	Pin	Muncul peringatan bahwa mahasiswa hanya dapat mengambil satu ide skripsi.		Berhasil
Ide Skripsi	Deskripsi	Dosen PJ	Kuantitas	Action																										
Sistem Ide Skripsi	Membuat perencanaan dan implementasi sistem ide skripsi satu login satu database	Chapko Pabichah	2	Pin																										
Sistem perubahan bentuk	Website sistem perubahan bentuk	Hario Setjawan	1	Pin																										
Sistem Penggajian Prodi IT	Sistem yang memudahkan penggajian dosen dan karyawan di Prodi IT	Azzam	1	Pin																										
Sistem Pencarian Kuliab	Membuatkan 3 Mahasiswa Login Database HCI	Hario Setjawan	1	Pin																										

4.4 Pembahasan

Website menyediakan fasilitas *login* untuk tiga level pengguna, yaitu administrator, Dosen, dan Mahasiswa. Pada tiap level pengguna, sistem menyediakan fungsi-fungsi yang hanya dapat diakses oleh pengguna dengan levelnya masing-masing.

Pada *login* level administrator, sistem menyediakan menu manajemen akun mahasiswa dan dosen. Dalam manajemen akun mahasiswa dan dosen, Administrator dapat melihat, mencari dan mengurutkan data akun. Administrator juga dapat melakukan reset password akun mahasiswa dan dosen, serta menghapus akun mahasiswa dan dosen.

Pada *login* level dosen, sistem menyediakan menu profil, data ide skripsi, dan seleksi mahasiswa. Pada menu profil, dosen dapat mengubah data diri. Menu data ide skripsi menampilkan data ide skripsi dosen yang bersangkutan. Pada menu ide skripsi, dosen dapat melakukan pencarian, pengurutan, penambahan, perubahan, dan penghapusan data ide skripsi. Menu seleksi mahasiswa menampilkan data mahasiswa yang mengambil ide skripsi. Pada menu seleksi mahasiswa, dosen dapat memilih mahasiswa untuk kemudian memberikan keputusan diterima atau tidak diterima.

Pada *login* level mahasiswa, sistem menyediakan menu profil, dimana mahasiswa dapat mengubah data diri. Selain itu, terdapat fasilitas pengambilan ide skripsi. Mahasiswa dapat melihat daftar ide skripsi pada halaman ide skripsi, kemudian mengambil ide skripsi yang diinginkan. Jika mahasiswa sudah mengambil ide skripsi, mahasiswa bisa melihat pengumuman untuk mengetahui hasil seleksi.

Secara keseluruhan, sistem dapat berjalan dengan baik sesuai dengan tujuan penulis dalam membuat *Website* Pengambilan Ide Skripsi. Sistem dapat membantu mahasiswa dalam mendapatkan informasi ide skripsi dari dosen. Mahasiswa juga bisa mengambil ide skripsi melalui sistem. Dosen pun dapat mengumumkan hasil seleksi mahasiswa pengambil ide skripsi.