**BAB IV  
HASIL DAN PEMBAHASAN**

1. **Spesifikasi Perangkat Keras**

Perangkat keras atau *hardware* merupakan peralatan fisik dari sistem komputer. *Hardware* merupakan salah satu perangkat yang sangat penting, karena jika tanpa *hardware* yang memenuhi syarat maka sebuah sistem aplikasi yang dibangun tidak dapat berjalan.

Adapun perangkat keras yang digunakan peneliti dalam penerapan Sistem Informasi Geografis ini adalah komputer dengan spesifikasi sebagai berikut :

* 1. *Processor* Intel® Pentium® CPU B960 @ 2.20GHz
  2. RAM 2.00 GB
  3. *Hardisk* 250 GB
  4. Monitor dengan resolusi 1366 x 768 *pixels*
  5. Terkoneksi dengan *internet*

1. **Spesifikasi Perangkat Lunak**

Selain perangkat keras juga dibutuhkan perangkat lunak yang tidak kalah pentingnya agar sistem ini dapat berjalan. Perangkat lunak merupakan istilah umum untuk data yang diformat dan disimpan secara digital, termasuk program komputer, dokumentasinya, serta berbagai informasi yang dibaca dan ditulis oleh komputer.

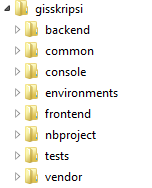
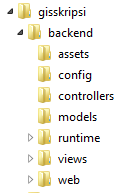
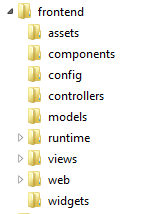
Dalam penerapan Sistem Informasi Geografis ini, telah dianalisis perangkat lunak yang digunakan adalah dengan spesifikasi sebagai berikut :

* + 1. Sistem operasi *Microsoft* *Windows* 8.1
    2. *Microsoft* *office* 2013
    3. MySQL 5.0.10 sebagai basis data
    4. ESRI® ArcMapTM 10.0 untuk pengembang sistem
    5. *Leaflet* yang merupakan *Javascript* *library* untuk pemetaan intraktif.
    6. Google Chrome sebagai *web* *browser*.
    7. Notepad++ sebagai *text* *editor*.
    8. Yii *framework* versi 2.0.
    9. XAMPP v.1.8.2 dengan termasuk didalamnya PHP versi 5.4.22, Apache 2.4.7 dan Tomcat 7.0.42 sebagai Java *server*.
    10. GeoServer 2.5.1 sebagai aplikasi untuk menyimpan data spasial

1. **Implementasi Bisnis Proses Usulan**

Sistem Informasi Geografis ini dibangun dengan menggunakan bantuan Yii *Framework* versi 2 *advanced*. Sehingga pemprogramannya menggunakan konsep *Model* *View* *Controller* (MVC) serta terpisah antara pengkodean untuk *frontend* dan pengkodean *banckend*. Sedangkan editor yang digunakan adalah notepad++.

Struktur penyimpanan folder dapat dilihat pada gambar dibawah ini:

Gambar ?? Struktur folder aplikasi

Berikut merupakan implementasi dari proses-proses yang telah diusulkan pada sebelumnya:

**Proses** **menampilkan peta dasar *default***

Dalam proses menampilkan peta dasar dalam *web* aplikasi ini dibantu dengan library *javascript leaflet.* Untuk menampilkan peta dasar sebagai tampilan *default* menggunkan fungsi *initilizez* yang dipanggil di *tag html body* dengan *event onload.* Berikut potongan *script*nya:

**function** initializez**()**

**{**

map **=** L**.**map**(**'map'**).**setView**([-**1.889306**,**114.697266**],** 4**);**

**var** osmUrl**=**'http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png'**;**

**var** osmAttrib**=**'BPS RI'**;**

**var** osm2 **=** **new** L**.**TileLayer**(**osmUrl**,** **{**minZoom**:** 0**,** maxZoom**:** 13**,** attribution**:** osmAttrib **}).**addTo**(**map**);**

**var** defaultLayer **=** L**.**tileLayer**.**provider**(**'Esri.WorldTopoMap'**).**addTo**(**map**);**

**var** baseLayers **=** **{**

'Thunderforest Landscape'**:** L**.**tileLayer**.**provider**(**'Thunderforest.Landscape'**),**

'MapQuest Aerial'**:** L**.**tileLayer**.**provider**(**'MapQuestOpen.Aerial'**),**

'Stamen Watercolor'**:** L**.**tileLayer**.**provider**(**'Stamen.Watercolor'**),**

'Esri WorldStreetMap'**:** L**.**tileLayer**.**provider**(**'Esri.WorldStreetMap'**),**

'Esri DeLorme'**:** L**.**tileLayer**.**provider**(**'Esri.DeLorme'**),**

'Esri WorldImagery'**:** L**.**tileLayer**.**provider**(**'Esri.WorldImagery'**),**

'Esri WorldTerrain'**:** L**.**tileLayer**.**provider**(**'Esri.WorldTerrain'**),**

'Esri WorldShadedRelief'**:** L**.**tileLayer**.**provider**(**'Esri.WorldShadedRelief'**),**

'Esri WorldPhysical'**:** L**.**tileLayer**.**provider**(**'Esri.WorldPhysical'**),**

'Esri OceanBasemap'**:** L**.**tileLayer**.**provider**(**'Esri.OceanBasemap'**),**

'Esri NatGeoWorldMap'**:** L**.**tileLayer**.**provider**(**'Esri.NatGeoWorldMap'**),**

'Esri WorldGrayCanvas'**:** L**.**tileLayer**.**provider**(**'Esri.WorldGrayCanvas'**),**

'Acetate'**:** L**.**tileLayer**.**provider**(**'Acetate'**)**

**};**

L**.**control**.**layers**(**baseLayers**,{**collapsed**:** **true}).**addTo**(**map**);**

**var** skala **=** L**.**control**.**scale**({**position**:**'bottomright'**}).**addTo**(**map**);**

compas**.**addTo**(**map**);**

info**.**addTo**(**map**);**

warnalegenda**();}**

**Proses** **pemilihan tipe wilayah**

Pada proses ini aplikasi akan menampilakan dropdown yang berisi tipe wilayah. Berikut adalah script untuk menampilkan dropdown tipe wilayah:

<?php

$form=ActiveForm::begin();

$tipeWilayah= **new** \common\models\TipeWilayah();

**echo** $form->field($tipeWilayah, 'nama')->widget(Select2::classname(), [

'data' => ArrayHelper::map($tipeWilayah::find()->all(), 'id', 'nama'),

'options' => ['placeholder' => 'Tipe Wilayah'],

'pluginOptions' => [

'allowClear' => **true**

],

])->label('');

?>

**Proses pemilihan wilayah**

Proses ini merupakan *depend* dari proses pemilihan tipe wilayah. Berikut adalah *script* menampilkan *dropdown* nama wilayah beserta fungsi di dalam *site* *controller* untuk *depend* tipe wilayahnya:

$wilayah= **new** \common\models\Wilayah();

**echo** $form->field($wilayah, 'nama')->widget(DepDrop::classname(), [

'type' => DepDrop::TYPE\_SELECT2,

'select2Options'=>['pluginOptions'=>['allowClear'=>**true**]],

'pluginOptions'=>[

'depends'=>['tipewilayah-nama'],

'url' => Url::to(['/site/child-wilayah']),

'loadingText' => 'Loading Wilayah...',

'placeholder' => 'Nama Wilayah'

]

])->label('');

**public** **function** actionChildWilayah() {

\Yii::$app->response->format = \yii\web\Response::FORMAT\_JSON;

$out = [];

**if**(**isset**($\_POST['depdrop\_parents'])&&end($\_POST['depdrop\_parents'])!=**null**) {

$id = end($\_POST['depdrop\_parents']);

$query = **new** Query;

$query->select('id,nama')

->from('wilayah')

->where('tipe='.$id);

$list = $query->all();

$selected = **null**;

**if** ($id != **null** && **count**($list) > 0) {

$selected = '';

**foreach** ($list **as** $i => $account) {

$out[] = ['id' => $account['id'], 'name' => $account['nama']];

**if** ($i == 0) {

$selected = $account['id'];

}

}

**return** (['output' => $out, 'selected'=>$selected]);

}

}

**return** (['output' => '', 'selected'=>'']);

}

**Proses pemilihan topik**

Proses pemilihan topik merupakan *depend* terhadap pemilihan nama wilayah. Dropdown pemilihan topik akan menghasilkan *option* jika di daerah terpilih memiliki data. Di bawah ini merupakan *script*nya:

$topik= **new** \common\models\Topik();

**echo** $form->field($topik, 'nama')->widget(DepDrop::classname(), [

'type' => DepDrop::TYPE\_SELECT2,

'select2Options'=>['pluginOptions'=>['allowClear'=>**true**]],

'pluginOptions'=>[

'depends'=>['wilayah-nama'],

'url' => Url::to(['/site/child-topik']),

'loadingText' => 'Loading Topik..',

'placeholder' => 'Topik'

]

])->label('');

**public** **function** actionChildTopik() {

\Yii::$app->response->format = \yii\web\Response::FORMAT\_JSON;

$out = [];

**if**(**isset**($\_POST['depdrop\_parents'])&&end($\_POST['depdrop\_parents'])!=**null**) {

$id = end($\_POST['depdrop\_parents']);

$query = **new** Query;

$query->select('topik.id, topik.nama')

->from('fakta')

->distinct('topik.id')

->join('INNER JOIN', 'wilayah','fakta.id\_wilayah=wilayah.id')

->join('INNER JOIN', 'variabel','fakta.id\_variabel=variabel.id')

->join('INNER JOIN', 'topik','variabel.id\_topik=topik.id')

->orderBy('topik.nama')

->where('wilayah.id\_parent='.$id);

$list = $query->all();

$selected = **null**;

**if** ($id != **null** && **count**($list) > 0) {

$selected = '';

**foreach** ($list **as** $i => $topik) {

$out[] = ['id' => $id.$topik['id'], 'name' => $topik['nama']];

**if** ($i == 0) {

$selected = $id.$topik['id'];

}

}

**return** (['output' => $out, 'selected'=>$selected]);

}

}

**return** (['output' => '', 'selected'=>'']);

}

**Proses pemilihan variabel**

Proses pemilihan variabel merupakan *depend* terhadap pemilihan topik. Dropdown pemilihan variabel akan menghasilkan *option* jika di daerah terpilih memiliki data. Di bawah ini merupakan *script*nya**:**

**public** **function** actionChildVariabel() {

\Yii::$app->response->format = \yii\web\Response::FORMAT\_JSON;

$out = [];

**if**(**isset**($\_POST['depdrop\_parents'])&&end($\_POST['depdrop\_parents'])!=**null**) {

$id = end($\_POST['depdrop\_parents']);

$l=strlen($id)-10;

$idTop=substr($id,10,$l);

$idWil=substr($id,0,10);

$query = **new** Query;

$query->select('variabel.id, variabel.nama')

->from('fakta')

->distinct('variabel.id')

->join('INNER JOIN', 'wilayah','fakta.id\_wilayah=wilayah.id')

->join('INNER JOIN', 'variabel','fakta.id\_variabel=variabel.id')

->join('INNER JOIN', 'topik','variabel.id\_topik=topik.id')

->orderBy('variabel.nama')

->where('topik.id='.$idTop.' AND wilayah.id\_parent='.$idWil);

$list = $query->all();

$selected = **null**;

**if** ($id != **null** && **count**($list) > 0) {

$selected = '';

**foreach** ($list **as** $i => $variabel) {

$out[] = ['id' => $idWil.$variabel['id'], 'name' => $variabel['nama']];

**if** ($i == 0) {

$selected = $idWil.$variabel['id'];

}

}

**return** (['output' => $out, 'selected'=>$selected]);

}

}

**return** (['output' => '', 'selected'=>'']);

}

$variabel= **new** \common\models\Variabel();

**echo** $form->field($variabel, 'nama')->widget(DepDrop::classname(), [

'type' => DepDrop::TYPE\_SELECT2,

'select2Options'=>['pluginOptions'=>['allowClear'=>**true**]],

'pluginOptions'=>[

'depends'=>['topik-nama'],

'url' => Url::to(['/site/child-variabel']),

'loadingText' => 'Loading Variabel ...',

'placeholder' => 'Variabel'

]

])->label('');

**Proses pemilihan kategori**

Proses pemilihan kategori merupakan *depend* terhadap pemilihan variabel. Dropdown pemilihan kategori akan menghasilkan *option* jika di daerah terpilih memiliki data. Di bawah ini merupakan *script*nya

**public** **function** actionChildKategori() {

\Yii::$app->response->format=\yii\web\Response::FORMAT\_JSON;

$out = [];

**if**(**isset**($\_POST['depdrop\_parents'])&&end($\_POST['depdrop\_parents'])!=**null**) {

$id = end($\_POST['depdrop\_parents']);

$l=strlen($id)-10;

$idVar=substr($id,10,$l);

$idWil=substr($id,0,10);

$query = **new** Query;

$query->select('kategori.id, kategori.nama')

->from('fakta')

->distinct('kategori.id')

->join('INNER JOIN', 'wilayah','fakta.id\_wilayah=wilayah.id')

->join('INNER JOIN', 'variabel','fakta.id\_variabel=variabel.id')

->join('INNER JOIN', 'kategori','fakta.id\_kategori=kategori.id')

->orderBy('kategori.nama')

->where('variabel.id='.$idVar.' AND wilayah.id\_parent='.$idWil);

$list = $query->all();

$selected = **null**;

**if** ($id != **null** && **count**($list) > 0) {

$selected = '';

**foreach** ($list **as** $i => $variabel) {

$out[] = ['id' => $variabel['id'], 'name' => $variabel['nama']];

**if** ($i == 0) {

$selected = $variabel['id'];

}

}

**return** (['output' => $out, 'selected'=>$selected]);

}

}

**return** (['output' => '', 'selected'=>'']);

}

$kategori= **new** \common\models\Kategori();

**echo** $form->field($kategori, 'nama')->widget(DepDrop::classname(), [

'type' => DepDrop::TYPE\_SELECT2,

'select2Options'=>['pluginOptions'=>['allowClear'=>**true**]],

'pluginOptions'=>[

'depends'=>['variabel-nama'],

'url' => Url::to(['/site/child-kategori']),

'loadingText' => 'Loading kategori ...',

'placeholder' => 'Kategori'

]

])->label('');

**Proses pemanggilan data**

Proses ini akan berjalan ketika *botton generate* di*click* oleh *user.* Fungsi calldata adalah sebuah fungsi *javascript* yang akan me*request* data dengan bantuan *ajax.* Berikut adalah potongan fungsi *javascript*nya:

**function** calldata**(**aWil**){**

loadingt**(**0**);**

**var** ta **=** **new** Array**();**

**var** idP **=** **new** Array**();**

**var** nP **=** **new** Array**();**

**var** bu**=** **new** Array**();**

indexTahun**=**0**;**

$**.**ajax**({**

url**:** '?r=site/data&wil='**+**aWil**+**'&var='**+**aVar**+**'&kat='**+**aKat**,**

type **:** 'POST'**,**

dataType **:** 'json'**,**

success**:** **function(**data**)**

**{**

idProvinsi.splice**(**0**,** idProvinsi.length**);**

tahun.splice**(**0**,** tahun.length**);**

**var** j **=** 0**;**

data.data.forEach**(function(**entry**)** **{**

//...

**Proses pemanggilan data shapefile**

Proses ini merupakan kelanjutan dari peroses pemanggilan data*.* Data shapefile yang disimpan di geoserver akan dipanggil mengunakan ajax request dengan output tipe json. Berikut adalah potongan fungsi *javascript*nya:

**function** callProvMap**(**aWil**)**

**{**

$**.**ajax**({**

url**:** '?r=geoserver-url/load-peta&idWil='**+**aWil**,**

type **:** 'POST'**,**

dataType **:** 'json'**,**

success**:** **function(**data**)**

**{**

kodewilayah**=**aWil**;**

**if(**aWil**!=**lokasi**[**posisi**])**

**{**lokasi**[(**posisi**+**1**)]** **=** aWil**;**

posisi**++;**

//...

**Proses pewarnaan peta**

Setelah data shapefile peta berhasil dipanggil maka proses berikutnya adalah pewarnaan peta. Proses ini akan menghasilkan peta yang telah diwarnai berdasakan kelasnya. Berikut adalah potongan fungsi *javascript*nya:

**function** setVariableTahun**(**valTahun**)** **{**

vVal**=**valTahun**;**

**var** jp**=new** Array**();**

**var** npr**=new** Array**();**

**var** i**=**0**;**

idProvinsi**.**forEach**(function(**untukSeries**)** **{**

npr**[**i**]=**namaProvinsi**[**untukSeries**];**

jp**[**i**]=**dataTabel**[**valTahun**][**untukSeries**];**

i**++;**

**});**

//...

**Proses *setting* lagenda**

Proses *setting* lagenda peta merupakan fasilitas bagi pengunjung untuk mengganti tipe kombinasi warna, jumlah kelas dan metode pengelompokan data . Proses ini bisa digunakan jika proses pewarnaan awal peta telah selesai. Berikut adalah potongan fungsi *javascript*nya:

**function** warnalegenda**(){**

//...

**var** $select **=** $**(**'<select></select>'**)**

**.**appendTo**(**$**(**'#metode'**))**

**.**on**(**'change'**,** **function()** **{**

noMetode**=**$**(this).**val**();**

setVariableTahun**(**tahun**[**indexTahun**]);**

**});**

**for** **(var** i **=** 0**;** i **<** metode.length**;** i**++)** **{**

$**(**'<option></option>'**)**

**.**text**(**metode**[**i**])**

**.**attr**(**'value'**,** i**)**

**.**appendTo**(**$select**);}**

$**(**'#myForm input'**).**on**(**'change'**,** **function()** **{**

noWarna **=** $**(**'input[name=warnaa]:checked'**,** '#myForm'**).**val**();**

setVariableTahun**(**tahun**[**indexTahun**]);});**

$**(**"#spin input"**).**bind**(**'keyup mouseup'**,** **function** **()** **{**

jlhKelas **=** $**(**'input[name=jmkelas]'**,** '#spin'**).**val**();**

setVariableTahun**(**tahun**[**indexTahun**]);});}**

**Proses menampilkan grafik dibawah peta**

Proses menampilkan grafik di bawah peta akan otomatis berjalan ketika pewarnaan peta telah selesai . Berikut adalah *script* fungsi *javascript*nya:

**function** grafikPeta**(**namawilayah**,**isdata**){**

$**(**'#grafikS'**).**html**(**'<canvas id="myChart" style="width: 100% !important;height: 300px !important;"></canvas>'**);**

**var** myNewChart**;**

**var** barChartData **=** **{**

labels **:** namawilayah**,**

datasets **:** **[**

**{**

fillColor **:** "rgba(151,187,205,0.5)"**,**

strokeColor **:** "rgba(151,187,205,0.8)"**,**

highlightFill **:** "rgba(151,187,205,0.75)"**,**

highlightStroke **:** "rgba(151,187,205,1)"**,**

data **:** isdata

**}**

**]**

**}**

**var** ctx **=** document.getElementById**(**"myChart"**).**getContext**(**"2d"**);**

myNewChart **=** **new** Chart**(**ctx**).**Bar**(**barChartData**,** **{**

//responsive : true,

animation**:** **false,**

**});**

**}**

**Proses menampilkan grafik ketika *polygon* peta diklik**

Berikut adalah *script* fungsi *javascript* untuk menampilan grafik ketika *polygon* di*click*:

**Implementasi Basis Data**

**function** grafikPetaline**(**e**){**

**var** idWill**=** e.target.feature.properties.ID**;**

layerPopup **=** L.popup**({**maxWidth**:**700**})**

**.**setLatLng**(**e.latlng**)**

**.**setContent**(**'<span id="maxi" class="glyphicon glyphicon-resize-full" style="cursor: pointer" aria-hidden="true" onclick="maxzoom('**+**idWill**+**')"></span><span id="minimize" class="glyphicon glyphicon-resize-small" style="cursor: pointer" aria-hidden="true" onclick="minimize('**+**idWill**+**')"></span><h5>'**+**namaProvinsi**[**idWill**]+**' '**+**tahun**[**0**]+**'-'**+**tahun**[**tahun.length**-**1**]+**'</h5><div id="lineChart"></div><div id="zoomIn"></div>'**)**

**.**openOn**(**map**);**

cekData**(**idWill**);**

$**(**'#minimize'**).**css**(**'display'**,**'none'**);**

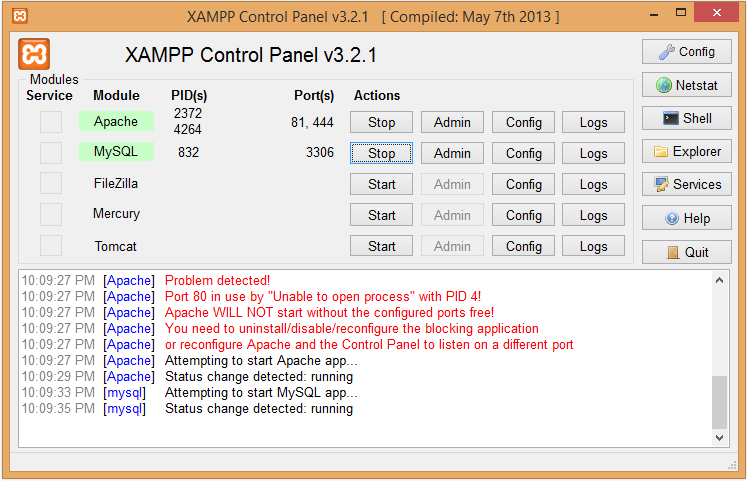
chartpop**(**idWill**);**

**}**

Berdasarkan hasil perancangan basis data yang telah dibuat pada bab sebelumnya, kemudian pada bab ini akan dijelaskan bagaimana pengimplementasian basis data tersebut.

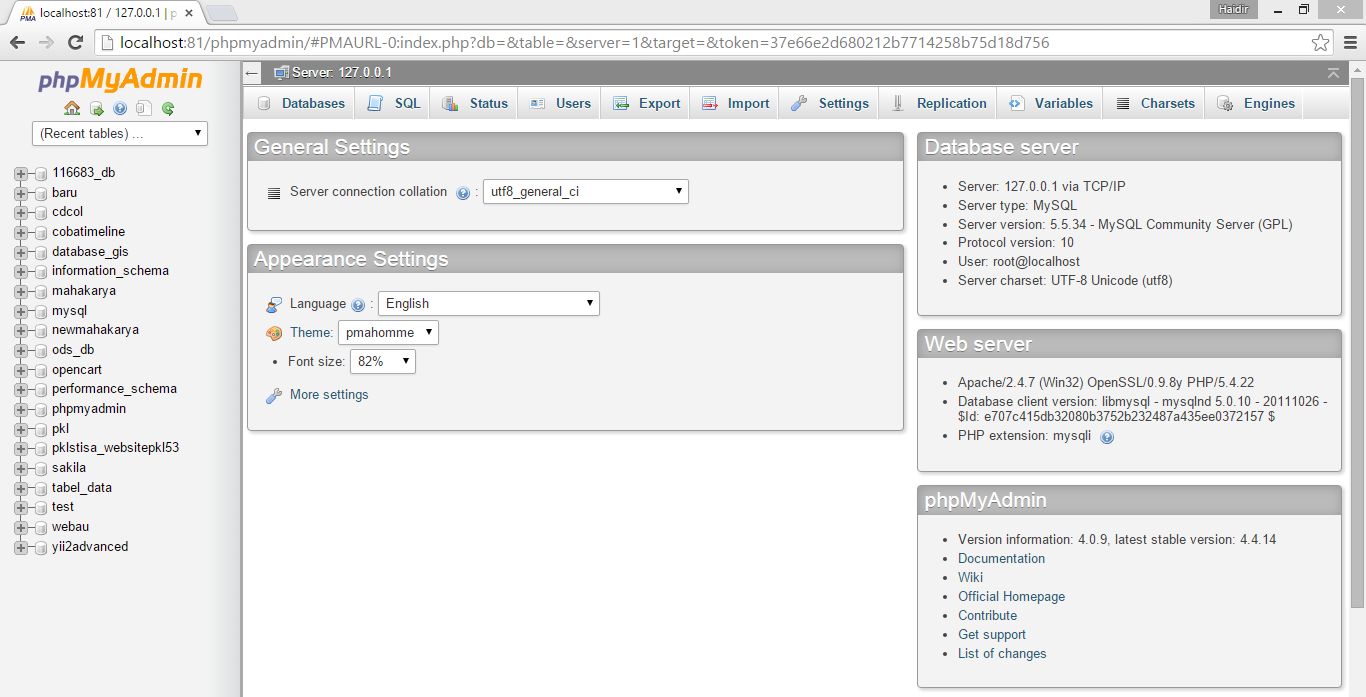
Implementasi basis data jenis pertama yaitu data atribut. Implementasi data atribut pada aplikasi ini menggunakan XAMPP v.1.8.2 serta MySQL 5.0.10 sebagai alat antarmuka basis data editornya. Sedangkan implementasi basis data jenis kedua yaitu data peta *shapefile* menggunakan aplikasi *geoserver.* Berikut merupakan tahapan implementasi:

1. Data atribut
   1. Mengaktifkan XAMPP *service* yang telah diinstal di komputer. Dapat dilihat pada gambar *screenshoot* di bawah ini.



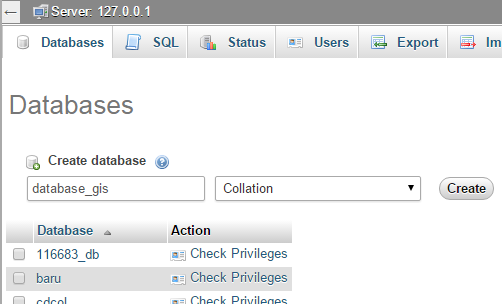
Gambar ?? Tampilan XAMPP control panel

* 1. Membuka phpMyAdmin pada browser dengan alamat : <http://localhost:81/phpmyadmin/>



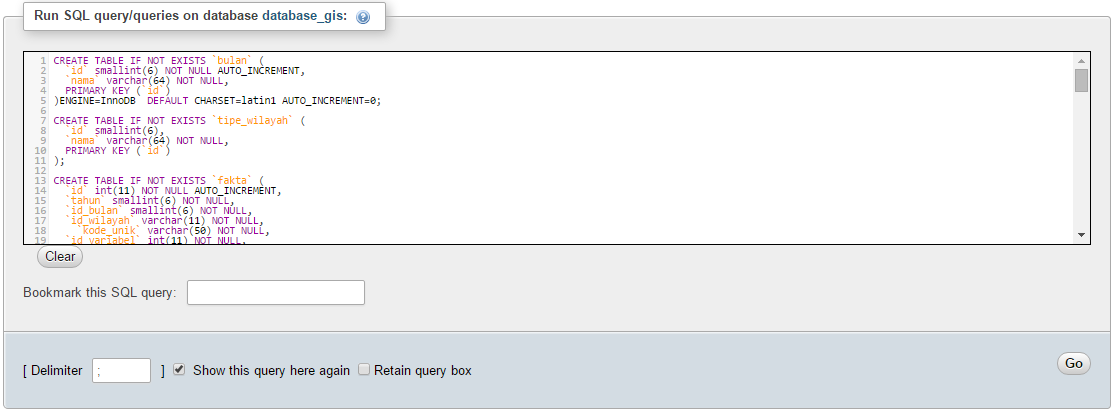
Gambar ?? Tampilan awal phpMyAdmin

* 1. Membuat nama database baru



Gambar ?? Tampilan membuat database baru

* 1. Membuat *query sql* pada tab *sql*



Gambar ?? Tampilan menjalankan *query sql*

Implementasi tabel bulan

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `bulan` **(**

`id` **smallint(**6**)** **NOT** **NULL** AUTO\_INCREMENT**,**

`nama` **varchar(**64**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`id`**)**

**)**ENGINE**=**InnoDB **DEFAULT** CHARSET**=**latin1 AUTO\_INCREMENT**=**0**;**

Implementasi tabel tipe\_wilayah

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `tipe\_wilayah` **(**

`id` **smallint(**6**),**

`nama` **varchar(**64**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`id`**)**

**);**

Implementasi tabel fakta beserta relasinya

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `fakta` **(**

`id` **int(**11**)** **NOT** **NULL** AUTO\_INCREMENT**,**

`tahun` **smallint(**6**)** **NOT** **NULL,**

`id\_bulan` **smallint(**6**)** **NOT** **NULL,**

`id\_wilayah` **varchar(**11**)** **NOT** **NULL,**

`kode\_unik` **varchar(**50**)** **NOT** **NULL,**

`id\_variabel` **int(**11**)** **NOT** **NULL,**

`id\_kategori` **int(**11**)** **NOT** **NULL,**

`id\_item\_kategori` **int(**11**)** **NOT** **NULL,**

`id\_sumber\_data` **int(**11**)** **NOT** **NULL,**

`nilai` **double** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`id`**),**

**KEY** `id\_bulan` **(**`id\_bulan`**),**

**KEY** `id\_wilayah` **(**`id\_wilayah`**),**

**KEY** `id\_variabel` **(**`id\_variabel`**),**

**KEY** `id\_kategori` **(**`id\_kategori`**),**

**KEY** `id\_item\_kategori` **(**`id\_item\_kategori`**),**

**KEY** `id\_sumber\_data` **(**`id\_sumber\_data`**)**

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**latin1 AUTO\_INCREMENT**=**1 **;**

**ALTER** **TABLE** `fakta` **ADD** **UNIQUE** **(**`kode\_unik`**);**

**ALTER** **TABLE** `fakta` **ADD** **CONSTRAINT** `fakta\_id\_bulan` **FOREIGN** **KEY** **(**`id\_bulan`**)** **REFERENCES** `database\_gis`**.**`bulan` **(**`id`**)** **ON** **DELETE** **CASCADE,** **ADD** **CONSTRAINT** `fakta\_id\_wilayah` **FOREIGN** **KEY** **(**`id\_wilayah`**)** **REFERENCES** `database\_gis`**.**`wilayah` **(**`id`**)** **ON** **DELETE** **CASCADE,** **ADD** **CONSTRAINT** `fakta\_id\_variabel` **FOREIGN** **KEY** **(**`id\_variabel`**)** **REFERENCES** `database\_gis`**.**`variabel` **(**`id`**)** **ON** **UPDATE** **CASCADE,ADD** **CONSTRAINT** `fakta\_id\_kategori` **FOREIGN** **KEY** **(**`id\_kategori`**)** **REFERENCES** `database\_gis`**.**`kategori` **(**`id`**)** **ON** **DELETE** **CASCADE,** **ADD** **CONSTRAINT** `fakta\_id\_item\_kategori` **FOREIGN** **KEY** **(**`id\_item\_kategori`**)** **REFERENCES** `database\_gis`**.**`item\_kategori` **(**`id`**)** **ON** **DELETE** **CASCADE,** **ADD** **CONSTRAINT** `fakta\_id\_sumber\_data` **FOREIGN** **KEY** **(**`id\_sumber\_data`**)** **REFERENCES** `database\_gis`**.**`sumber\_data` **(**`id`**)** **ON** **DELETE** **CASCADE;**

Implementasi tabel item\_kategori beserta relasinya

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `item\_kategori` **(**

`id` **int(**11**)** **NOT** **NULL** AUTO\_INCREMENT**,**

`nama` **varchar(**255**)** **NOT** **NULL,**

`item\_no` **int(**11**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`id`**)**

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**latin1 AUTO\_INCREMENT**=**0**;**

Implementasi tabel kategori beserta relasinya

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `kategori` **(**

`id` **int(**11**)** **NOT** **NULL** AUTO\_INCREMENT**,**

`nama` **varchar(**64**)** **NOT** **NULL,**

`id\_variabel` **int(**11**)** **NOT** **NULL,**

`keterangan` text**,**

**PRIMARY** **KEY** **(**`id`**),**

**KEY** `id\_variabel` **(**`id\_variabel`**)**

**)**ENGINE**=**InnoDB **DEFAULT** CHARSET**=**latin1 AUTO\_INCREMENT**=**0**;**

**ALTER** **TABLE** `kategori` **ADD** **CONSTRAINT** `kategori\_id\_variabel` **FOREIGN** **KEY** **(**`id\_variabel`**)** **REFERENCES** `database\_gis`**.**`variabel` **(**`id`**)** **ON** **DELETE** **CASCADE;**

Implementasi tabel sumber\_data beserta relasinya

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `sumber\_data` **(**

`id` **int(**11**)** **NOT** **NULL** AUTO\_INCREMENT**,**

`tipe` **smallint(**6**),**

`nama\_cs` **varchar(**200**),**

`tanggal\_cs` **date,**

`institusi\_cs` text**,**

`deskripsi\_cs` text**,**

`nama\_buku` **varchar(**200**),**

`tanggal\_buku` **date,**

`penerbit\_buku` text**,**

`status` tinyint**(**1**),**

**PRIMARY** **KEY** **(**`id`**)**

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**latin1 AUTO\_INCREMENT**=**1**;**

Implementasi tabel topik beserta relasinya

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `topik` **(**

`id` **int(**11**)** **NOT** **NULL** AUTO\_INCREMENT**,**

`nama` **varchar(**64**)** **NOT** **NULL,**

`keterangan` text**,**

`id\_parent` **int(**11**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`id`**)**

**)**ENGINE**=**InnoDB **DEFAULT** CHARSET**=**latin1 AUTO\_INCREMENT**=**1**;**

Implementasi tabel variabel beserta relasinya

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `variabel` **(**

`id` **int(**11**)** **NOT** **NULL** AUTO\_INCREMENT**,**

`id\_topik` **int(**11**)** **NOT** **NULL,**

`nama` **varchar(**64**)** **NOT** **NULL,**

`keterangan` text**,**

`satuan` **varchar(**64**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`id`**),**

**KEY** `id\_topik` **(**`id\_topik`**)**

**)**ENGINE**=**InnoDB **DEFAULT** CHARSET**=**latin1 AUTO\_INCREMENT**=**1**;**

**ALTER** **TABLE** `variabel` **ADD** **CONSTRAINT** `variabel\_id\_topik` **FOREIGN** **KEY** **(**`id\_topik`**)** **REFERENCES** `database\_gis`**.**`topik` **(**`id`**)** **ON** **DELETE** **CASCADE;**

Implementasi tabel wilayah beserta relasinya

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `wilayah` **(**

`id` **varchar(**11**)** **NOT** **NULL,**

`nama` **varchar(**64**)** **NOT** **NULL,**

`id\_parent` **varchar(**11**)** **NOT** **NULL,**

`tipe` **int(**11**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`id`**)**

**KEY** `tipe` **(**`tipe`**)**

**);**

**ALTER** **TABLE** ` wilayah` **ADD** **CONSTRAINT** `wilayah\_tipe` **FOREIGN** **KEY** **(**`tipe`**)** **REFERENCES** `database\_gis`**.**`tipe\_wilayah` **(**`id`**)** **ON** **DELETE** **CASCADE;**

Implementasi tabel geoserver\_url beserta relasinya

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `geoserver\_url` **(**

`id` **int(**11**)** **NOT** **NULL** AUTO\_INCREMENT**,**

`id\_wilayah` **varchar(**11**)** **NOT** **NULL,**

`url` text **NOT** **NULL,**

`zoom` **smallint(**3**)** **NOT** **NULL,**

`center\_x` **double** **NOT** **NULL,**

`center\_y` **double** **NOT** **NULL,**

`tipe` **int(**11**)** **NOT** **NULL,**

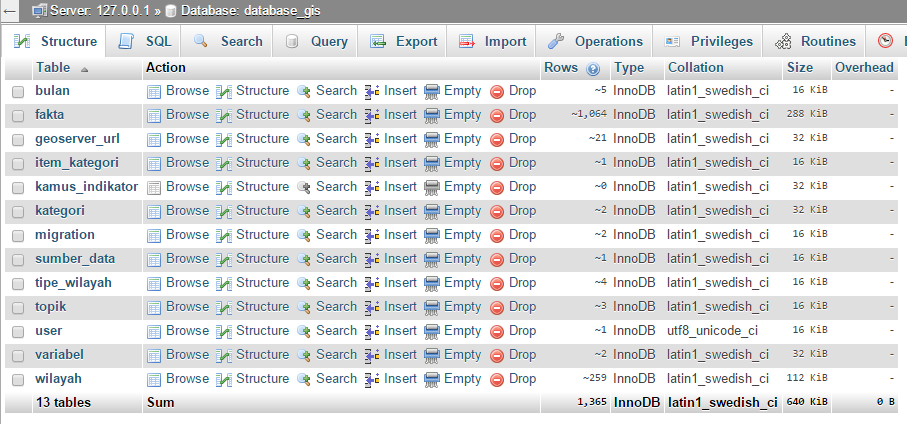
**PRIMARY** **KEY** **(**`id`**),**

**KEY** `id\_wilayah` **(**`id\_wilayah`**)**

**)**ENGINE**=**InnoDB **DEFAULT** CHARSET**=**latin1 AUTO\_INCREMENT**=**1**;**

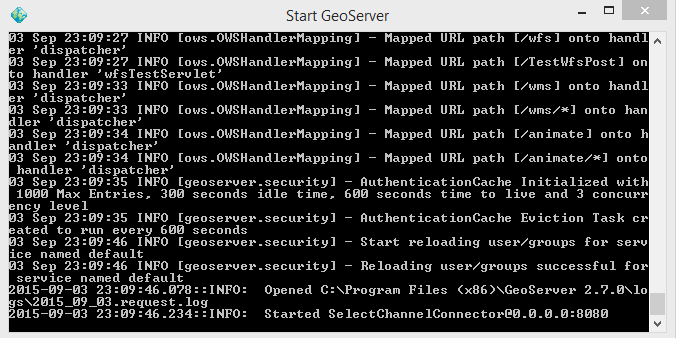
**ALTER** **TABLE** `geoserver\_url` **ADD** **CONSTRAINT** `geoserver\_url\_id\_wilayah` **FOREIGN** **KEY** **(**`id\_wilayah`**)** **REFERENCES** `database\_gis`**.**`wilayah` **(**`id`**)** **ON** **DELETE** **CASCADE;**

Berikut hasil dari pembuatan tabel-tabelnya.



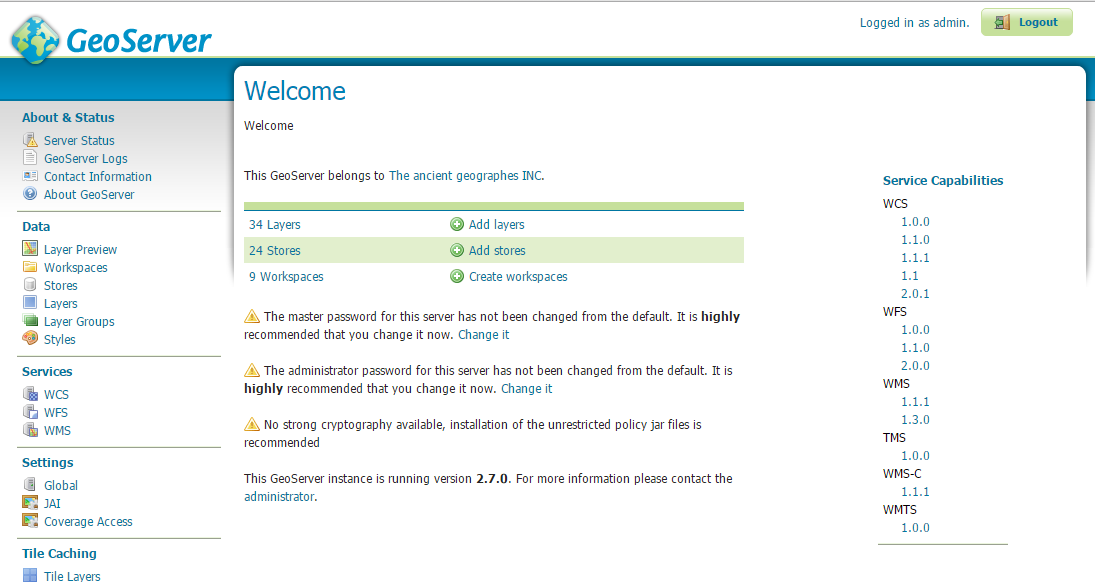
Gambar ?? Tampilan hasil pembuat tabel database

1. Data spasial
   1. Mengaktifkan geoserver yang sudah diinstal



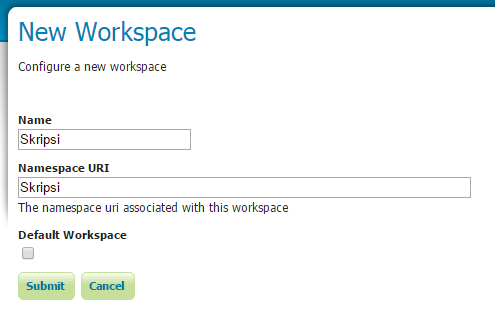
Gambar ?? Tampilan cmd geoserver yang sudah aktif

* 1. Mengakses geoserver menggunakan *browser* dengan *url*: <http://localhost:8080/geoserver/web> lalu loggin



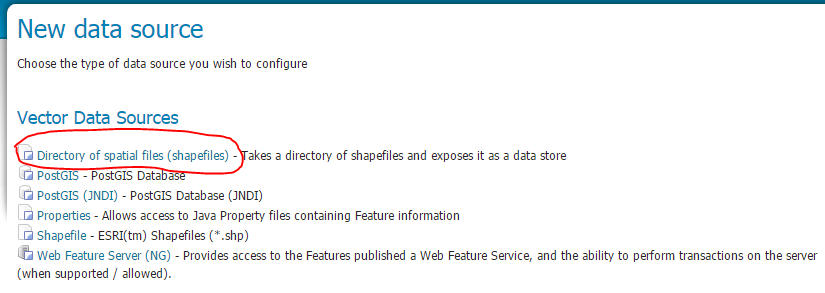
Gambar ?? Tampilan halaman awal *geoserver*

* 1. Membuat *workspace* baru



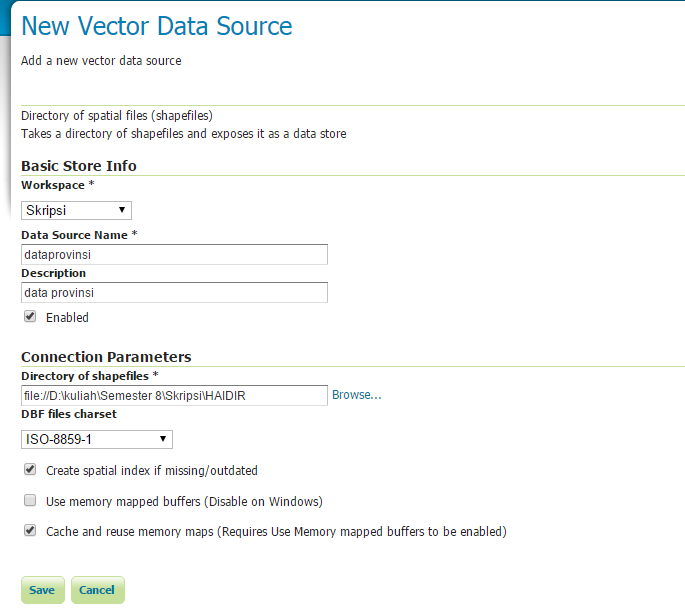
Gambar ?? Tampilan membuat *workspase* baru

* 1. Membuat *strore* baru di dalam *workspace* yang dibuat tadi



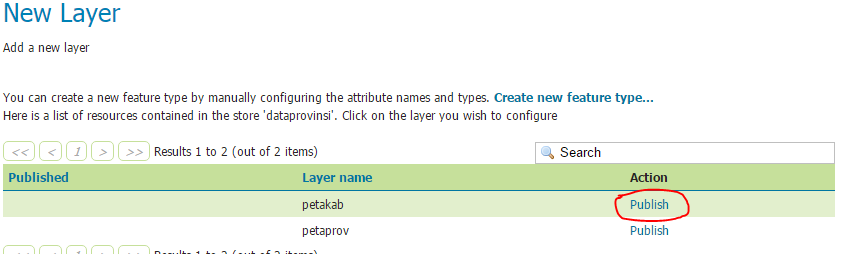
Gambar ?? Memilih tipe data *source*

* 1. Membuat nama *store* dan memilih alamat peta *shapefile* pada folder di dalam komputer.

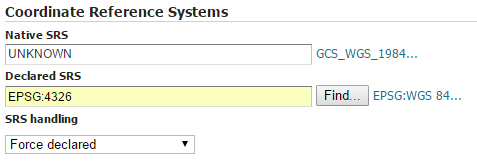


Gambar ?? Membuat *store* baru

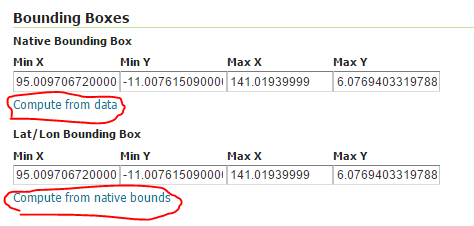
* 1. Mempublish *layer* yang sudah tersimpan di *strore*



Gambar ?? Publish layer

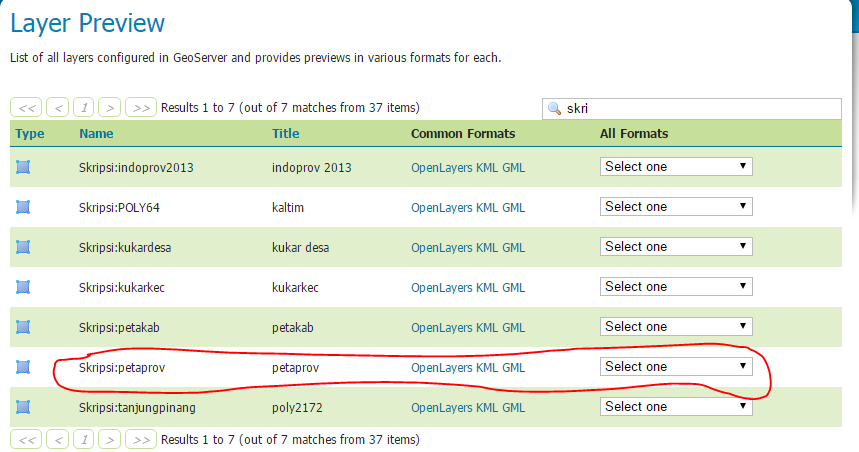


Gambar ?? pemilihan *coordinate reference system*



Gambar ?? Menghitung batas *shapefile*

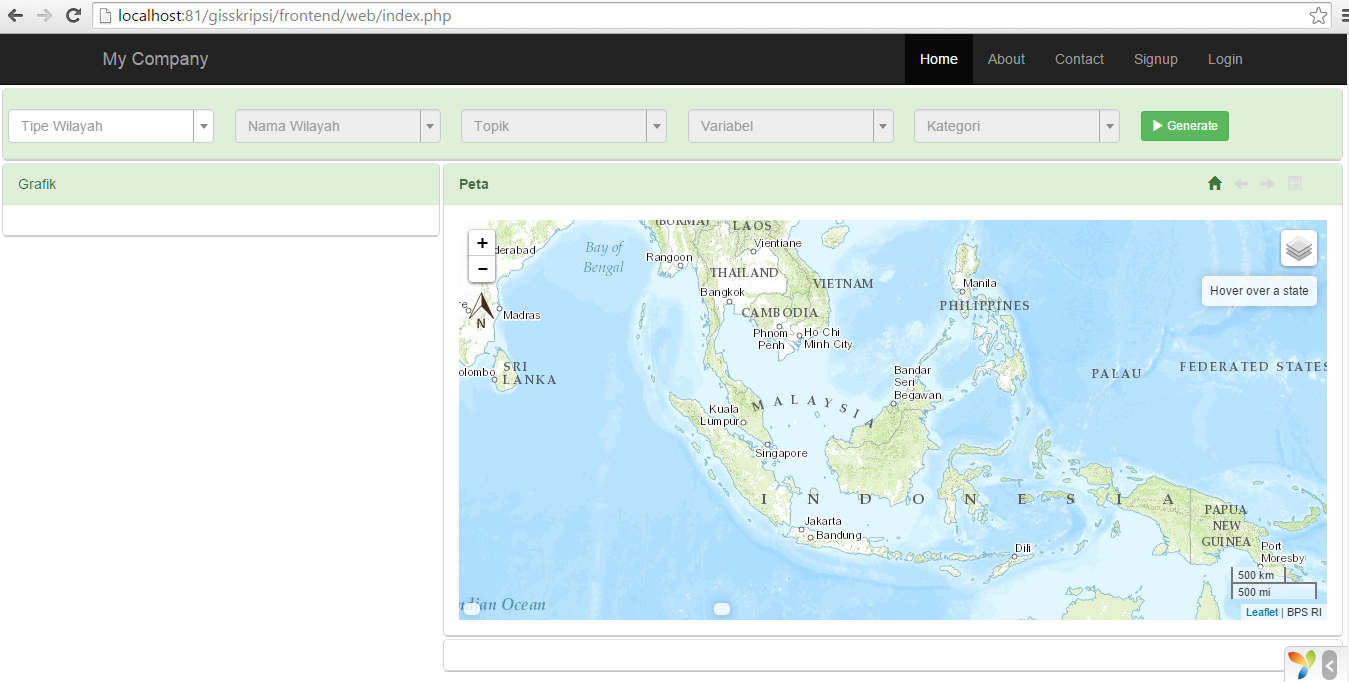
Hasil dari penyimpanan *layer* dapat dilihat di *layer* *preview*



Gambar ?? Tampilan *layer* *preview*

1. **Implementasi Antarmuka**

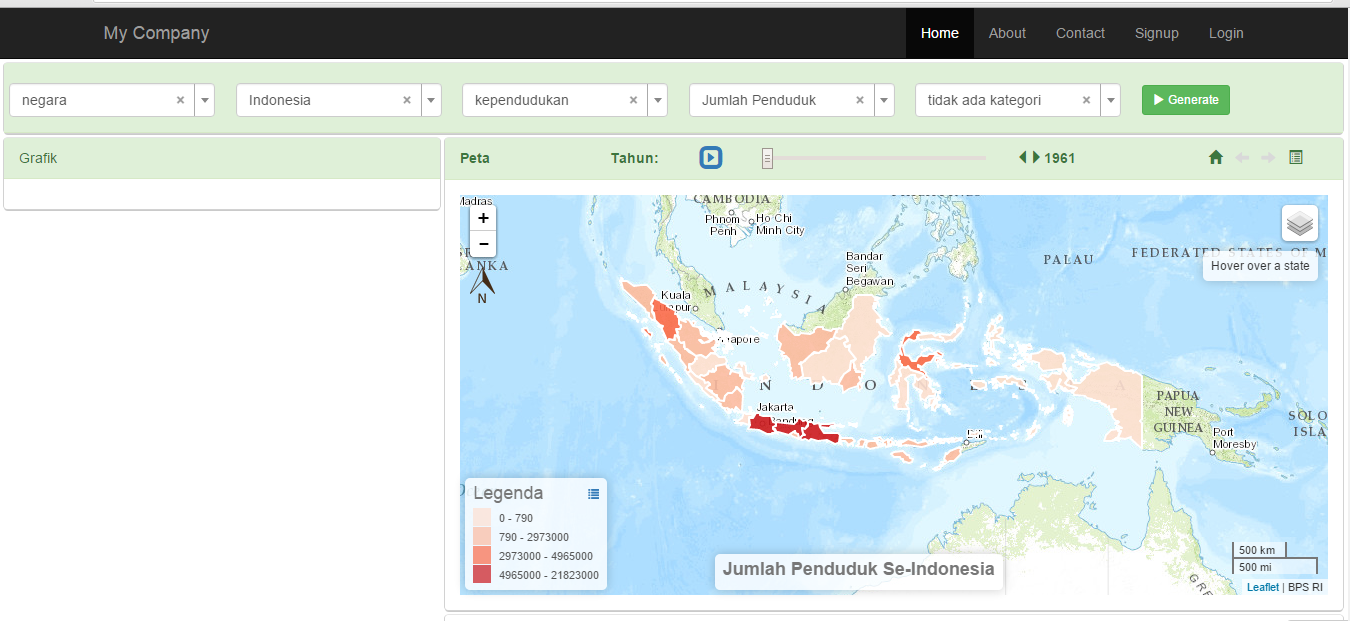
Antarmuka berfungsi sebagai media komunikasi antara pengguna dengan aplikasi. Implementasi antarmuka sistem usulan berdasarkan dengan rancangan antar muka yang telah dibuat di bab tiga. Berikut adalah hasil implementasi antarmuka yang sudah dibangun.



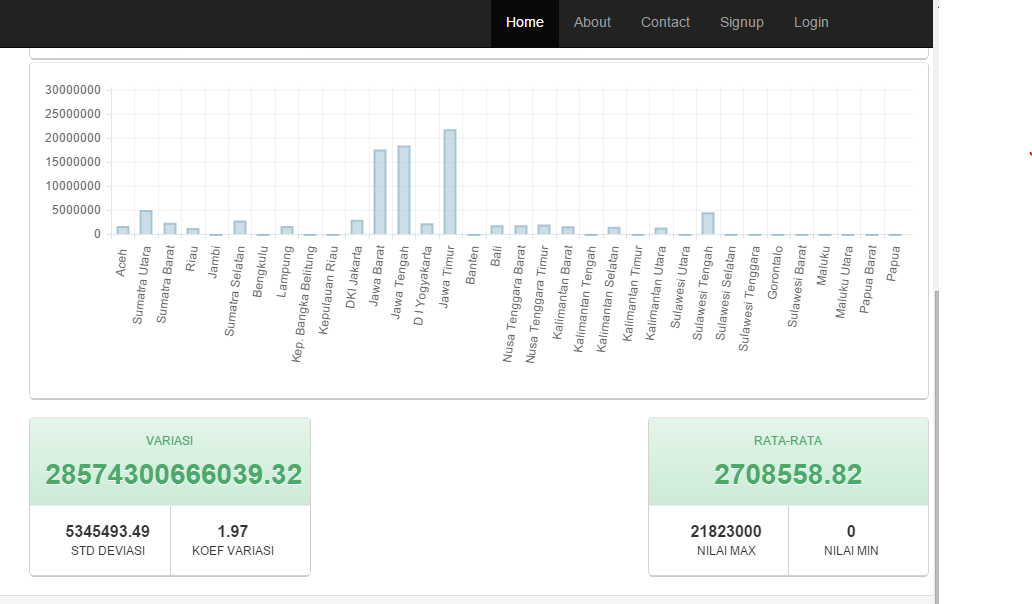
Gambar ?? Tampilan antarmuka *default* pengunjung

Gambar di atas adalah tampilan awal dari aplikasi untuk pengunjung. Pengunjung akan diberikan kebebasan untuk memilih wilayah dan data apa yang akan ditampilkan ke dalam peta tematik. Untuk menggunakannya pengunjung tinggal memilih pada *dropdown* yang telah disediakan.

Jika pengunjung sudah menentukan wilayah dan data yang ingin ditampilkan, kemudian pengunjung hanya cukup untuk mengklik tombol *generate.* Setelah itu aplikasi akan secara otomatis mengasilkan peta tematik sesuai dengan pemilihannya *dropdown*nya tadi. Berikut adalah contoh tampilan antarmuka ketika peta tematik telah di *generate* .

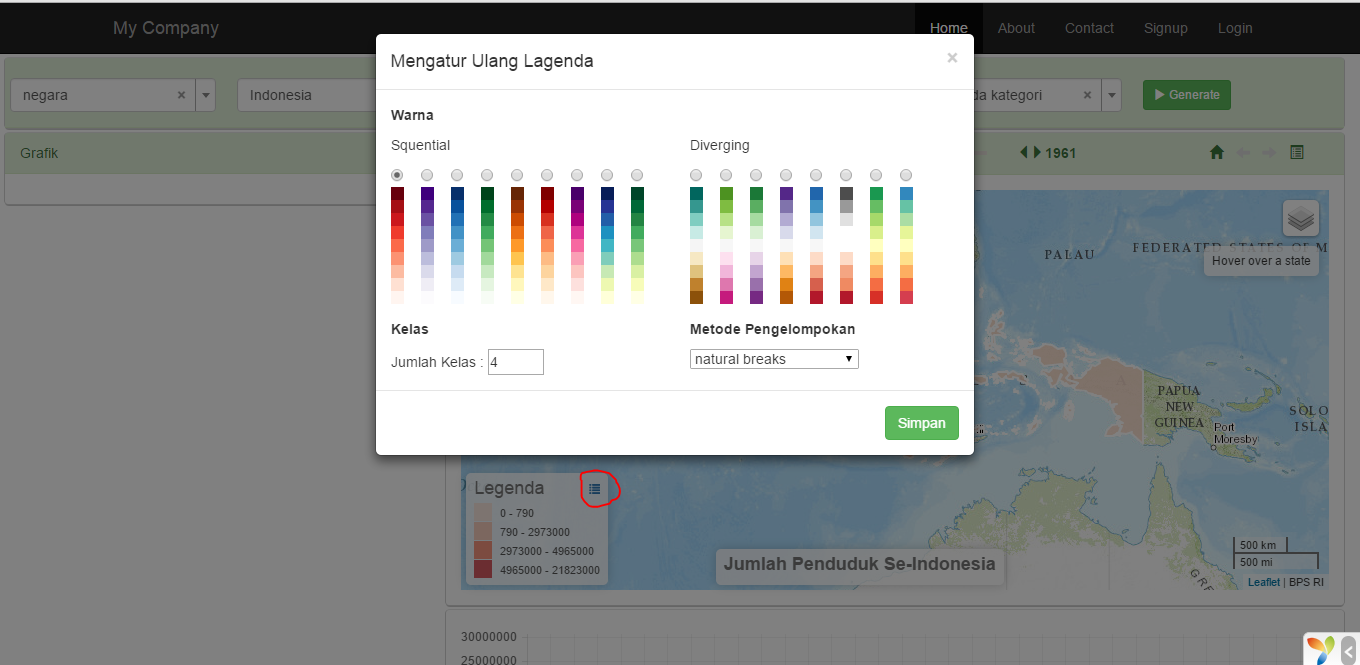


Gambar ?? Tampilan antarmuka peta tematik



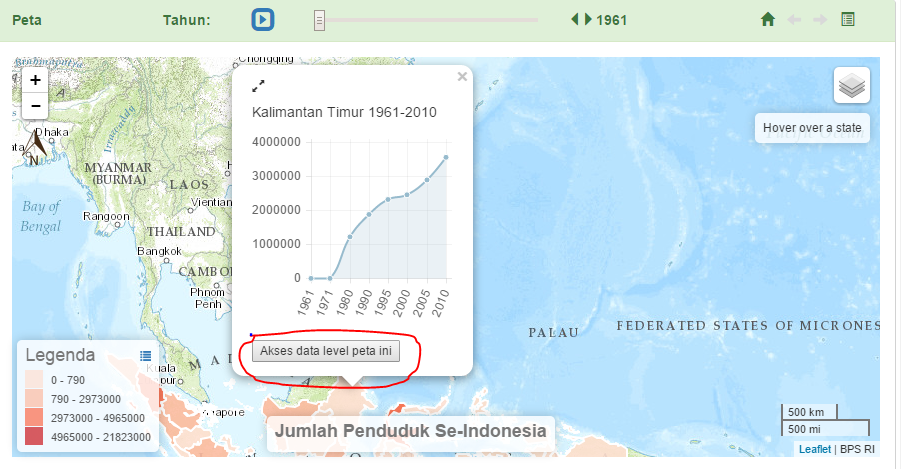
Gambar ?? Tampilan antarmuka grafik dan statistik di bawah peta

Setelah peta tematik berhasil ditampilkan pengunjung dapat mengubah pengaturan lagenda sesuai dengan kebutuhan, seperti pemilihan warna, banyak kelas, dan metode pengelompokan kelas dengan cara mengklik logo yang berada dipojok lagenda.



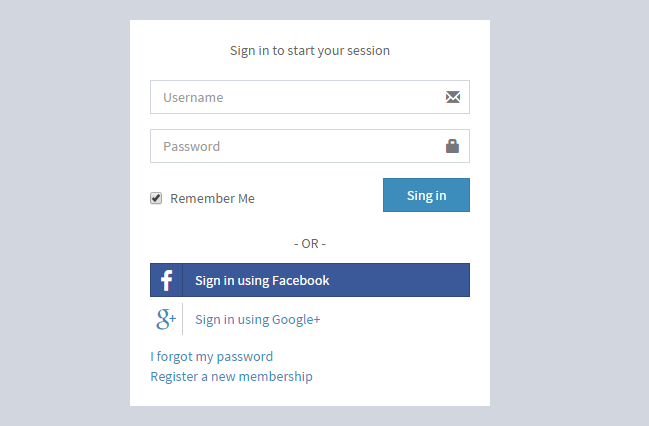
Gambar ?? Tampilan pengaturan lagenda

Jika polygon pada wilayah tertentu diklik maka akan menampikan popup berupa grafik *line* untuk data setiap periode waktunya. Jika di level polygon tersebut memiliki data dengan judul yang sama maka akan ada *botton* untuk mengakses data peta level tersebut. Lebih jelasnya dapat dilihat pada gambar dibawah ini.



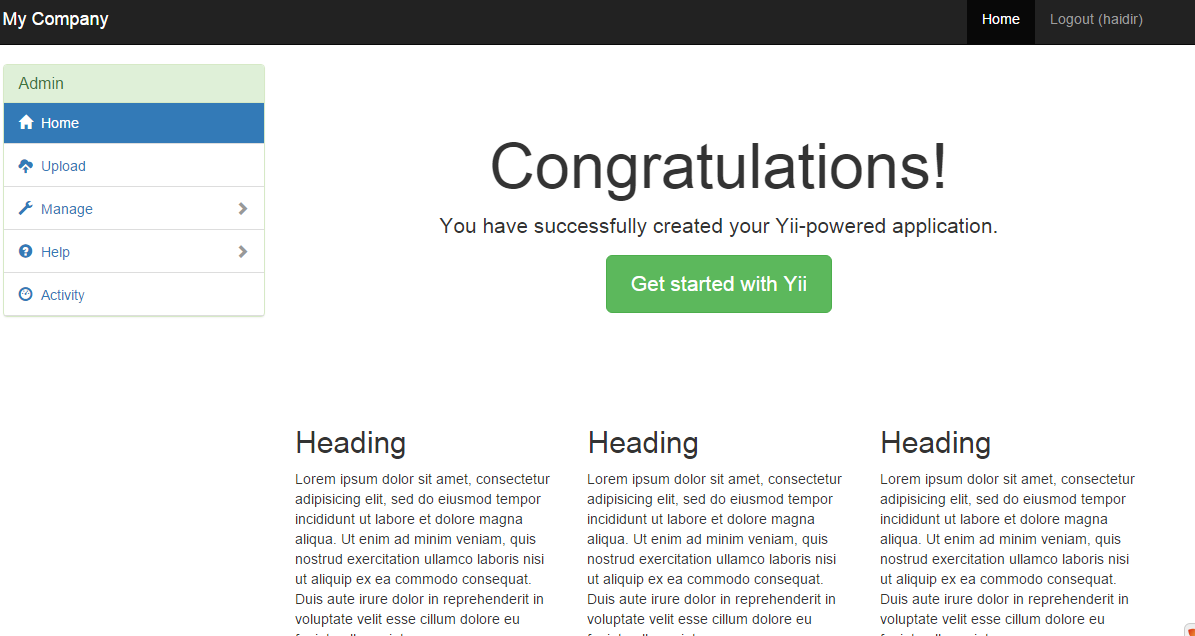
Gambar ?? Tampilan popup polygon ketika diklik

Sedangkan untuk tampilan antarmuka backend dapat diakses dengan url: <http://localhost:81/gisskripsi/backend/web/index.php>. Sebelum memasuki aplikasi ini maka akan admin akan diminta untuk memsukan username dan password.



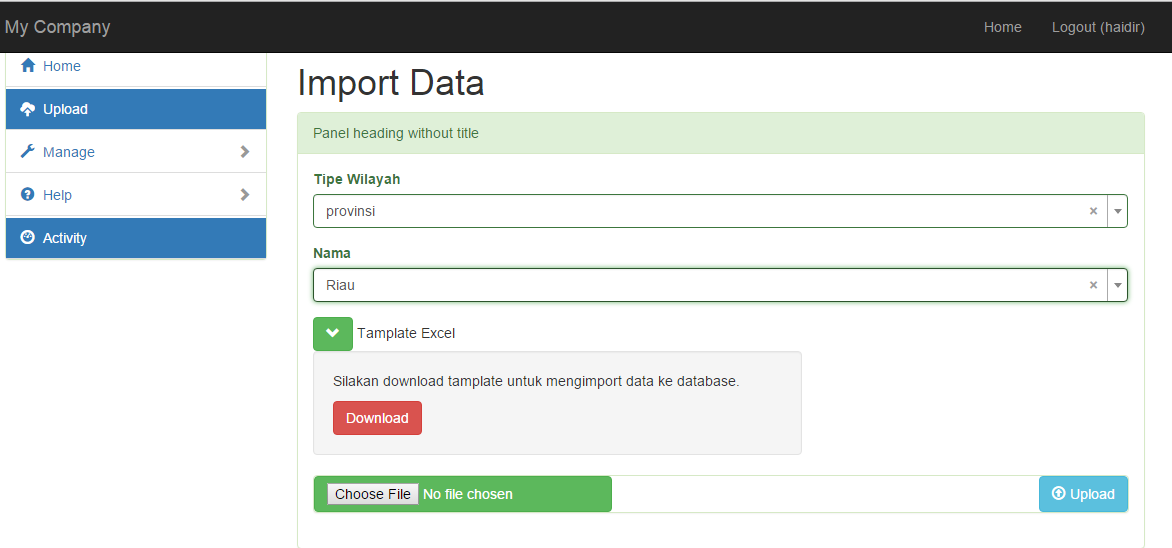
Gambar ?? Tampilan antarmuka untuk loggin

Setelah admin berhasil loggin maka aplikasi secara otomatis akan menuju halaman *home* khusus admin.



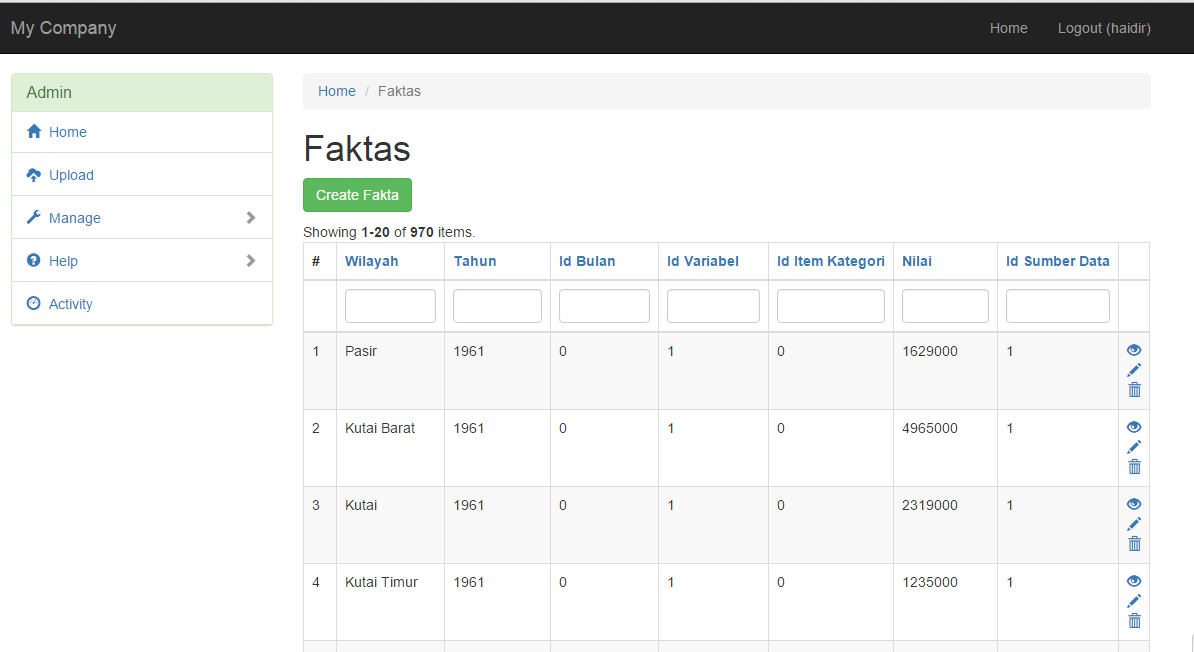
Gambar ?? Tampilan *home* admin

Didalam *backend* aplikasi ini terdapat menu untuk mengupload data spasial yang ingin dipublikasikan. Sebelum mengupload data spasial aplikasi telah menyediakan *template* berupa file dengan ekstensi xls yang dapat didownload berdasarkan nama wilayah yang ingin di*publish* datanya. Dapat dilihat pada gambar dibawah ini.



Gambar ?? Tampilan backend untuk mem*publish* data

Setelah data diunggah maka aplikasi secara otomatis akan menuju halaman *manage* fakta yang berarti halaman ini berfungsi untuk mengelola data yang sudah kita upload.



Gambar ?? Tampilan untuk mengelola data yang sudah diupload.

1. **Implementasi Jaringan dan Keamanan Data**

**Keamanan**

Berikut merupakan implementasi dari sisi keamanan data pada aplikasi ini:

1. generatePasswordHash: adalah sebuah metode untuk mengenkripsi password. Dimana metode ini akan menghasilkan string unik yang akan disimpam ke database. Untuk implementasinya fungsi ini ditempatkan didalam yii\base\Security.php yang di implements oleh model User.php.

**public** **function** setPassword($password)

{

$this->password\_hash = Yii::$app->security->generatePasswordHash($password);

}

1. validatePassword: Merupakan method yang menjadi pendamping dari generatePasswordHash. Method ini berfungsi sebagai pemeriksa password user apakah benar atau tidak.

**public** **function** validatePassword($password)

{

**return** Yii::$app->security->validatePassword($password, $this->password\_hash);

}

1. *Access Control Filter.* Fungsi ini digunakan untuk mengatur hak akses pengguna terhadap fungsi-fungsi yang terdapat di dalam aplikasi. Untuk implementasinya diletakan didalam *controller* yang akan dibatasi. Dibawah ini merupakan contoh implementasinya didalam siteController.php pada backend.
2. **Hasil Pengujian**

**public** **function** behaviors()

{

**return** [

'access'=>[

'class'=>AccessControl::className(),

'rules'=>[

[

'actions'=>['login','error'],

'allow'=>**true**,

],

[

'actions'=>['logout','index','download','upload','child'],

'allow'=>**true**,

'roles'=>['@'],

],

],

],

'verbs'=>[

'class'=>VerbFilter::className(),

'actions'=>[

'logout'=>['post'],

],

],

];

}

Setelah dilakukan implementati maka tahap selanjutnya adalah pengujian. Pengujian bertujuan untuk mengetahui apakah aplikasi berserta fungsi-fungsinya sudah berjalan sesuai dengan kebutuhan dan tujuan yang diinginkan. Selain itu juga dapat melihat kekurangan-kekurangan yang terdapat didalam aplikasi.

Pengujian ini akan mengunakan tiga pendekatan yaitu dengan teknik *white-box*, *black-box* dan *System Usability Scale* (SUS).

1. *White-box*

Pengujian dengan pendekatan *white-box* merupakan teknik pengujian aplikasi dengan menganalisis secara langsung kode-kode yang berada didalam aplikasi. Tujuan dari pengujian ini adalah untuk melihat apakah ada kesalahan kode atau tidak. Sehingga jika terdapat kesalahan yang membuat *output*nya tidak sesuai maka akan langsung diperbaiki dan di*compile* ulang.

* Uji instruksi dan percabangan

Pada uji ini yang diambil contoh adalah fungsi cekData. Fungsi cekData merupakan fungsi yang digunakan di dalam fungsi grafikPetaline. Fungsi cekData bertugas untuk mengecek data pada level tertentu apakah ada atau tidak. Jika peta di level polygon yang di klik memiliki data dengan judul yang sama dengan peta awal maka akan ada tombol untuk mengakses akses data level tersebut. Sebaliknya jika pada level peta tersebut tidak memiliki data dengan judul yang sama dengan peta level sebelumnnya maka tombol tersebut tidak akan muncul.

**function** cekData**(**aWil**){**

$**.**ajax**({**

url**:** '?r=site/cekdata&wil='**+**aWil**+**'&var='**+**aVar**+**'&kat='**+**aKat**,**

type **:** 'POST'**,**

dataType **:** 'json'**,**

success**:** **function(**data**)**

**{**

**if(**data.data**==**1**)** $**(**'#zoomIn'**).**html**(**'<button onclick="calldata('**+**aWil**+**')">Akses data level peta ini</button>'**);**

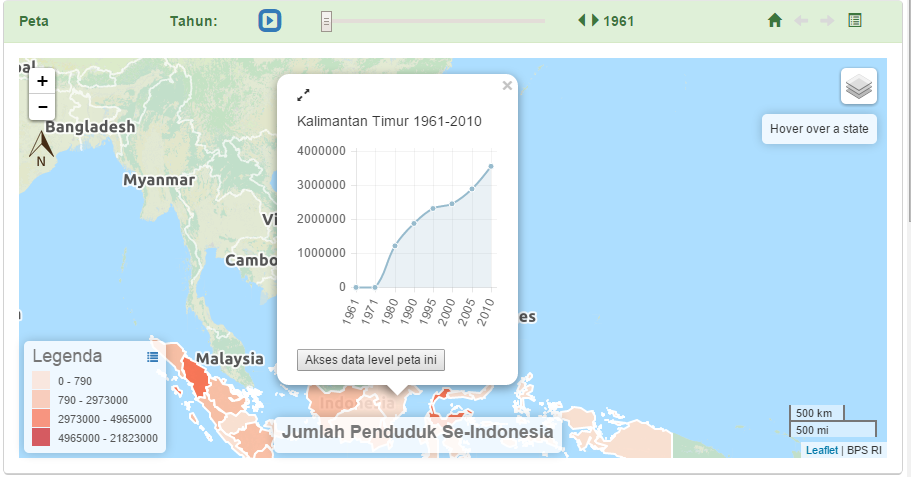
**else** $**(**'#zoomIn'**).**html**(**''**)**

**}**

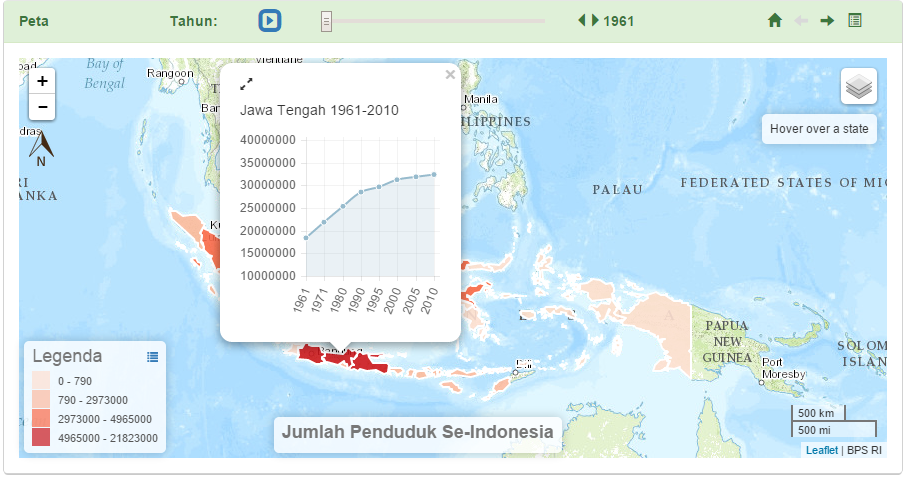
**});**

**}**

Pengujian untuk kasus ini menggunakan contoh *output* data pada lokasi Provinsi Kalimantan Timur dan Provinsi Jawa Tengah. Saat Provinsi Kaltim diklik akan ada tombol akses data yang muncul karena di Provinsi Kaltim memiliki data dengan judul yang sama dengan data Indonesia. Sedangkan pada Provinsi Jateng tidak ada muncul tombol akses data karena di provinsi tersebut tidak ada data dengan judul yang sama dengan data Indonesia.



Gambar ?? *Pupup* Provinsi Kaltim ketika terdapat data



Gambar ?? *Pupup* Provinsi Jawa Tengah ketika tidak memiliki data

* Pengujian perulangan

Pada uji ini yang akan dijadikan contoh adalah fungsi untuk menampilkan tabel dinamis. Fungsi ini bernama drawTable. Adapun kodenya adalah sebagai berikut:

Cara kerja fungsi ini adalah dengan membuat *tag html* menggunakan *javascript*. Sedangkan untuk perulangannya mengunakan *for* dua dimensi untuk membuat *element* baris dan kolom di dalam *tag* *table*. Di bawah ini merupakan tampilan dari output tabelnya.

**function** drawTable**()** **{**

$**(**'#myModalLabel'**).**html**(**judul**);**

$**(**'#untukTabel'**).**html**(**'<span class="glyphicon glyphicon-list-alt" style="cursor: pointer" aria-hidden="true" data-toggle="modal" data-target="#myModalTabel" data-toggle="tooltip" data-placement="top" title="Tampilkan Data"></span>'**);**

$**(**'#tabel-dinamis'**).**html**(**''**);**

$**(**'#tabel-dinamis'**).**append**(**'<thead id="tbhead"></thead>'**);**

$**(**'#tbhead'**).**append**(**'<tr id="barishead"></tr>'**);**

$**(**'#barishead'**).**append**(**'<th>Nama Wilayah</th>'**);**

**for** **(var** c **=** 0**;** c **<** tahun.length**;** c**++)** **{**

$**(**'#barishead'**).**append**(**'<th>'**+**tahun**[**c**]+**'</th>'**);**

**}**

$**(**'#tabel-dinamis'**).**append**(**'<tbody id="tbbody"></tbody>'**);**

**for** **(var** r **=** 0**;** r **<** idProvinsi.length**;** r**++)** **{**

$**(**'#tbhead'**).**append**(**'<tr id="barisbody'**+**r**+**'"></tr>'**);**

$**(**'#barisbody'**+**r**).**append**(**'<th style="font-weight: normal">'**+**namaProvinsi**[**idProvinsi**[**r**]]+**'</th>'**);**

**for** **(var** c **=** 0**;** c **<** tahun.length**;** c**++)** **{**

$**(**'#barisbody'**+**r**).**append**(**'<th style="font-weight: normal">'**+**dataTabel**[**tahun**[**c**]][**idProvinsi**[**r**]]+**'</th>'**);**

**}**

**}**

**}**



Gambar ?? Tampilan hasil perulangan dalam pembuatan tabel

1. *Black-box*
2. *System Usability Scale* (SUS)