

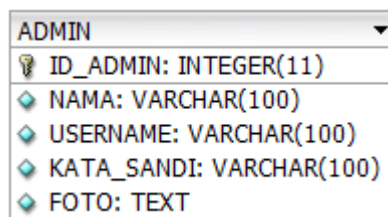
## BAB 5 IMPLEMENTASI

Pada sub bab ini akan dijelaskan mengenai proses implementasi dari analisis kebutuhan dan perancangan yang telah dibuat pada Bab 4. Pada sub bab ini penulis akan menjelaskan mengenai langkah-langkah pembuatan Sistem Informasi Pemetaan Kecelakaan Kota Batu yang telah diimplementasikan menggunakan bahasa pemrograman *PHP, html, css, dan Javascript*.

### 5.1 Implementasi *Database*

Pada tahap ini, mengimplementasikan *database* dengan cara membuat tabel yang mengacu pada rancangan *database* pada Gambar 4.24 pada Bab 4. Terdapat 5 tabel pada rancangan *database* yaitu ADMIN, DATA\_KECELAKAAN, JALAN, TITIK, dan WILAYAH. Berikut adalah implementasi tabel *database* yang telah dibuat pada perangkat pengolah *database MySQL*.

#### 5.1.1 Tabel ADMIN



ADMIN
ID_ADMIN: INTEGER(11)
NAMA: VARCHAR(100)
USERNAME: VARCHAR(100)
KATA_SANDI: VARCHAR(100)
FOTO: TEXT

**Gambar 5.1 Rancangan *Database* Tabel ADMIN**

Berdasarkan rancangan *database* tabel ADMIN pada Gambar 5.1 maka telah dibuat tabel *database* ADMIN dengan menggunakan *DDL (data definition language)* seperti yang dapat dilihat pada Tabel 5.1.

**Table 5.1 *DDL* Tabel Admin**

1	CREATE TABLE `admin` (
2	`ID_ADMIN` int(11) NOT NULL AUTO_INCREMENT PRIMARY KEY,
3	`NAMA` varchar(100) NOT NULL,
4	`USERNAME` varchar(100) NOT NULL,
5	`KATA_SANDI` varchar(100) NOT NULL,
6	`FOTO` text NOT NULL
7	)

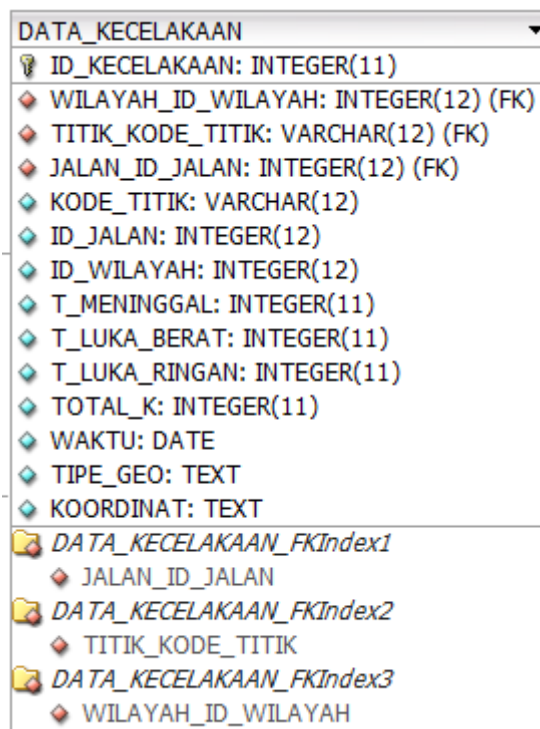
Berdasarkan *DDL* pada Tabel 5.1 didapatkan sebuah struktur *database* untuk tabel ADMIN yang bisa dilihat pada Gambar 5.2.

#	Name	Type	Collation	Attributes	Null	Default	Extra
<input type="checkbox"/>	1	ID_ADMIN	int(11)		No	None	AUTO_INCREMENT
<input type="checkbox"/>	2	NAMA	varchar(100)		No	None	
<input type="checkbox"/>	3	USERNAME	varchar(100)		No	None	
<input type="checkbox"/>	4	KATA_SANDI	varchar(100)		No	None	
<input type="checkbox"/>	5	FOTO	text		No	None	

**Gambar 5.2 Tabel ADMIN**

Tabel ini berfungsi untuk menyimpan data admin yang akan menggunakan sistem. Struktur tabel ADMIN adalah tabel yang memiliki 5 kolom yaitu ID\_ADMIN, NAMA, USERNAME, KATA\_SANDI, dan FOTO. Pada kolom ID\_ADMIN menggunakan *auto\_increment* untuk menghindari duplikasi data.

### 5.1.2 Tabel DATA\_KECELAKAAN



**Gambar 5.3 Rancangan Database Tabel Data Kecelakaan**

Berdasarkan rancangan *database* tabel DATA\_KECELAKAAN pada Gambar 5.3 maka telah dibuat tabel *database* DATA\_KECELAKAAN dengan menggunakan DDL seperti yang dapat dilihat pada Tabel 5.2.

**Table 5.2 DDL Tabel Data Kecelakaan**

1	CREATE TABLE `data_kecelakaan` (
2	`ID_KECELAKAAN` int(11) NOT NULL AUTO_INCREMENT PRIMARY KEY,

```

3      `KODE_TITIK` varchar(12) DEFAULT NULL,
4      `ID_JALAN` int(12) DEFAULT NULL,
5      `ID_WILAYAH` int(12) DEFAULT NULL,
6      `T_MENINGGAL` int(11) NOT NULL,
7      `T_LUKA_BERAT` int(11) NOT NULL,
8      `T_LUKA_RINGAN` int(11) NOT NULL,
9      `TOTAL_K` int(12) NOT NULL,
10     `WAKTU` date NOT NULL,
11     `TIPE_GEO` text NOT NULL,
12     `KOORDINAT` text NOT NULL,
13     FOREIGN KEY (`ID_JALAN`) REFERENCES `jalan` (`ID_JALAN`),
14     FOREIGN KEY (`KODE_TITIK`) REFERENCES `titik`
15     (`KODE_TITIK`),
16     FOREIGN KEY (`ID_WILAYAH`) REFERENCES `wilayah`
17     (`ID_WILAYAH`)
18 )

```

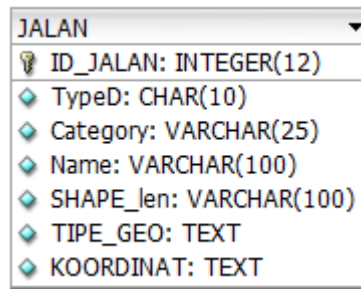
Berdasarkan DDL pada Tabel 5.2 didapatkan sebuah struktur *database* untuk tabel DATA KECELAKAAN yang bisa dilihat pada Gambar 5.4.

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	ID_KECELAKAAN	int(11)			No	None	AUTO_INCREMENT
2	KODE_TITIK	varchar(12)			Yes	NULL	
3	ID_JALAN	int(12)			Yes	NULL	
4	ID_WILAYAH	int(12)			Yes	NULL	
5	T_MENINGGAL	int(11)			No	None	
6	T_LUKA_BERAT	int(11)			No	None	
7	T_LUKA_RINGAN	int(11)			No	None	
8	TOTAL_K	int(12)			No	None	
9	WAKTU	date			No	None	
10	TIPE_GEO	text			No	None	
11	KOORDINAT	text			No	None	

**Gambar 5.4 Tabel Data Kecelakaan**

Tabel ini berfungsi untuk menyimpan data kecelakaan yang akan digunakan pada sistem. Struktur tabel DATA\_KECELAKAAN adalah tabel yang memiliki 11 kolom yaitu ID\_KECELAKAAN, KODE\_TITIK (*foreign key* dari tabel TITIK), ID\_JALAN (*foreign key* dari tabel JALAN), ID\_WILAYAH (*foreign key* dari tabel WILAYAH), T\_MENINGGAL, T\_LUKA\_BERAT, T\_LUKA\_RINGAN, TOTAL\_K, WAKTU, TIPE\_GEO, dan KOORDINAT.

### 5.1.3 Tabel JALAN



**Gambar 5.5 Rancangan Database Tabel Jalan**

Berdasarkan rancangan *database* tabel JALAN pada Gambar 5.5 maka telah dibuat tabel *database* JALAN dengan menggunakan seperti yang dapat dilihat pada Tabel 5.3.

**Table 5.3 DDL Tabel Jalan**

```

1 CREATE TABLE `jalan` (
2     `ID_JALAN` int(12) NOT NULL AUTO_INCREMENT PRIMARY KEY,
3     `TypeD` char(10) NOT NULL,
4     `Category` varchar(25) NOT NULL,
5     `Name` varchar(100) NOT NULL,
6     `SHAPE_len` varchar(100) NOT NULL,
7     `TIPE_GEO` text NOT NULL,
8     `KOORDINAT` text NOT NULL
9 )

```

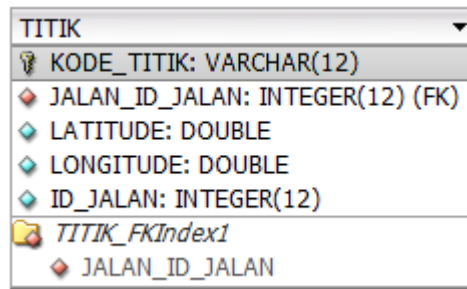
Berdasarkan *DDL* pada Tabel 5.3 didapatkan sebuah struktur *database* untuk tabel JALAN yang bisa dilihat pada Gambar 5.6.

#	Name	Type	Collation	Attributes	Null	Default	Extra
<input type="checkbox"/>	1 ID_JALAN	int(12)			No	None	AUTO_INCREMENT
<input type="checkbox"/>	2 TypeD	char(10)			No	None	
<input type="checkbox"/>	3 Category	varchar(25)			No	None	
<input type="checkbox"/>	4 Name	varchar(100)			No	None	
<input type="checkbox"/>	5 SHAPE_len	varchar(100)			No	None	
<input type="checkbox"/>	6 TIPE_GEO	text			No	None	
<input type="checkbox"/>	7 KOORDINAT	text			No	None	

**Gambar 5.6 Tabel Jalan**

Tabel ini berfungsi untuk menyimpan data spasial jalan dari hasil *export QGIS* ke dalam *database mysql*. Struktur tabel JALAN adalah tabel yang memiliki 7 kolom yaitu ID\_JALAN, TypeD, Category, Name, SHAPE\_len, TIPE\_GEO, dan KOORDINAT.

### 5.1.4 Tabel TITIK



**Gambar 5.7 Rancangan Database Tabel Titik**

Berdasarkan rancangan *database* tabel TITIK pada Gambar 5.7 maka telah dibuat tabel *database* TITIK dengan menggunakan *DDL* seperti yang dapat dilihat pada Tabel 5.4.

**Table 5.4 DDL Tabel Titik**

1	CREATE TABLE `titik` (
2	`KODE_TITIK` varchar(12) NOT NULL AUTO_INCREMENT PRIMARY
3	KEY,
4	`LATITUDE` double NOT NULL,
5	`LONGITUDE` double NOT NULL,
6	`ID_JALAN` int(12) NOT NULL,
7	FOREIGN KEY (`ID_JALAN`) REFERENCES `jalan` (`ID_JALAN`),
	)

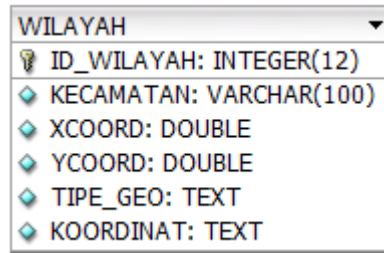
Berdasarkan *DDL* pada Tabel 5.4 didapatkan sebuah struktur *database* untuk tabel TITIK yang bisa dilihat pada Gambar 5.8.

#	Name	Type	Collation	Attributes	Null	Default	Extra
<input type="checkbox"/>	1 KODE_TITIK	varchar(12)			No	None	
<input type="checkbox"/>	2 LATITUDE	double			No	None	
<input type="checkbox"/>	3 LONGITUDE	double			No	None	
<input type="checkbox"/>	4 ID_JALAN	int(12)			No	None	

**Gambar 5.8 Tabel Titik**

Tabel ini berfungsi untuk menyimpan data spasial berupa koordinat baru lokasi kecelakaan. Struktur tabel TITIK adalah tabel yang memiliki 4 kolom yaitu KODE\_TITIK, LATITUDE, LONGITUDE, dan ID\_JALAN.

### 5.1.5 Tabel WILAYAH



**Gambar 5.9 Rancangan *Database* Tabel Wilayah**

Berdasarkan rancangan *database* tabel WILAYAH pada Gambar 5.9 maka telah dibuat tabel *database* WILAYAH dengan menggunakan *DDL* seperti yang dapat dilihat pada Tabel 5.5.

**Table 5.5 *DDL* Tabel Wilayah**

1	CREATE TABLE `wilayah` (
2	`ID_WILAYAH` int(12) NOT NULL,
3	`KECAMATAN` varchar(100) NOT NULL,
4	`XCOORD` double NOT NULL,
5	`YCOORD` double NOT NULL,
6	`TIPE_GEO` text NOT NULL,
7	`KOORDINAT` text NOT NULL
8	)

Berdasarkan *DDL* pada Tabel 5.5 didapatkan sebuah struktur *database* untuk tabel WILAYAH yang bisa dilihat pada Gambar 5.10.

#	Name	Type	Collation	Attributes	Null	Default	Extra
<input type="checkbox"/> 1	ID_WILAYAH	int(12)			No	None	
<input type="checkbox"/> 2	KECAMATAN	varchar(100)			No	None	
<input type="checkbox"/> 3	XCOORD	double			No	None	
<input type="checkbox"/> 4	YCOORD	double			No	None	
<input type="checkbox"/> 5	TIPE_GEO	text			No	None	
<input type="checkbox"/> 6	KOORDINAT	text			No	None	

**Gambar 5.10 Tabel Wilayah**

Tabel ini berfungsi untuk menyimpan data spasial batas wilayah dari hasil *export QGIS* ke dalam *database mysql*. Struktur tabel WILAYAH adalah tabel yang memiliki 6 kolom yaitu ID\_WILAYAH, KECAMATAN, XCOORD, YCOORD, TIPE\_GEO, dan KOORDINAT.

## 5.2 Implementasi Fungsi

Pada tahap ini, mengimplementasikan fungsi yang telah dirancang pada bab 4 dengan cara membuat kode program. Terdapat 4 contoh fungsi yang akan dijelaskan pada tahap ini. Berikut adalah *source code* dari implementasi fungsi yang telah dibuat.

### 5.2.1 Menambahkan Lokasi Kecelakaan

Proses menambahkan lokasi kecelakaan menggunakan fungsi dengan nama `tambah_koordinat` yang terdapat pada *controller* `c_kelola`. Pada fungsi `tambah_lokasi` ini dilakukan dengan cara membuat beberapa deklarasi untuk menampung data dari masukan admin. Setelah itu data dikirim ke *database* melalui *model* `m_kelola`. Pada tabel 5.6 merupakan *source code* dan penjelasan dari fungsi `tambah_koordinat`:

**Table 5.6 Source Code Tambah Koordinat Kecelakaan**

```
1 function tambah_koordinat()
2 {
3     $kode_titik = $this->input->post('kode_titik');
4     $latitude = $this->input->post('latitude');
5     $longitude = $this->input->post('longitude');
6     $id_jal = $this->input->post('id_jal');
7     $data = array(
8         'kode_titik' => $kode_titik,
9         'latitude' => $latitude,
10        'longitude' => $longitude,
11        'id_jalan' => $id_jal
12    );
12    $this->m_kelola->tambah_koordinat($data, 'titik');
13    redirect('c_kelola/kelola_koordinat');
14 }
```

Fungsi `tambah_koordinat` kecelakaan digunakan untuk menambahkan lokasi kecelakaan baru. Fungsi ini dimulai dengan mengumpulkan data-data yang dibutuhkan, yaitu kode titik, koordinat latitude, longitude dari lokasi kecelakaan, dan id jalan. Data terlebih dahulu ditampung menggunakan beberapa *variable*, antara lain `$kode_titik`, `$latitude`, `$longitude`, dan `$id_jal`. Setelah itu data yang sudah ditampung akan digabungkan ke dalam bentuk *array* dengan nama `$data`. Struktur data *array* disesuaikan dengan struktur *table* di *database* yang akan menyimpan data kecelakaan baru. Selanjutnya data *array* tersebut dikirimkan ke *model* `m_kelola`. Data berhasil disimpan dan dapat ditampilkan pada halaman Kelola Data Kecelakaan.

## 5.2.2 Menambahkan Data Kecelakaan

Pada proses menambahkan data kecelakaan, fungsi yang digunakan adalah fungsi `tambah_data` yang berada pada *controller* `c_kelola`. Pada fungsi menambahkan data kecelakaan ini dengan cara membuat beberapa deklarasi variabel untuk mendapatkan data dari masukan admin maupun dari *database*. Setelah data didapatkan maka data tersebut dikirimkan ke *database* melalui model `m_kelola`. Berikut pada tabel 5.7 merupakan *source code* dari fungsi `tambah_data`:

**Table 5.7 Source Code Fungsi Tambah Data Kecelakaan**

1	<code>function tambah_data()</code>
2	<code>{</code>
3	<code>    \$kode_titik = \$this-&gt;input-&gt;post('kode_titik');</code>
4	<code>    \$titik = \$this-&gt;m_kelola-&gt;ambil_koordinat(\$kode_titik);</code>
5	<code>    \$id_jalan = \$this-&gt;input-&gt;post('id_jalan');</code>
6	<code>    \$nama_jalan = \$this-&gt;input-&gt;post('nama_jalan');</code>
7	<code>    \$kecamatan = \$this-&gt;input-&gt;post('kecamatan');</code>
8	<code>    \$id_wilayah = \$this-&gt;m_kelola-&gt;ambil_wilayah(\$kecamatan);</code>
9	<code>    \$wilayah_id = \$id_wilayah[0]['id_wilayah'];</code>
10	<code>    \$meninggal = \$this-&gt;input-&gt;post('meninggal');</code>
11	<code>    \$luka_berat = \$this-&gt;input-&gt;post('luka_berat');</code>
12	<code>    \$luka_ringan = \$this-&gt;input-&gt;post('luka_ringan');</code>
13	<code>    \$total = \$this-&gt;input-&gt;post('total');</code>
14	<code>    \$waktu = \$this-&gt;input-&gt;post('waktu');</code>
15	<code>    \$tipe_geo = 'Point';</code>
16	<code>    \$koordinat = '['.\$titik[0]['longitude'].',</code> <code>    '.\$titik[0]['latitude'].']';</code>
17	<code>    \$data = array( 18</code>
19	<code>        'kode_titik' =&gt; \$kode_titik, 20</code>
21	<code>        'id_jalan' =&gt; \$id_jalan, 21</code>
22	<code>        'id_jalan' =&gt; \$id_jalan, 22</code>
23	<code>        't_meninggal' =&gt; \$meninggal, 23</code>
24	<code>        't_luka_berat' =&gt; \$luka_berat, 24</code>
25	<code>        't_luka_ringan' =&gt; \$luka_ringan, 25</code>
26	<code>        'total_k' =&gt; \$total, 26</code>
27	<code>        'waktu' =&gt; \$waktu, 27</code>
28	<code>        'tipe_geo' =&gt; \$tipe_geo, 28</code>
29	<code>        'koordinat' =&gt; \$koordinat 29</code>
30	<code>    );</code>
31	<code>    \$this-&gt;m_kelola-&gt;tambah_data (\$data,'data_kecelakaan');</code>
	<code>    redirect('c_kelola/lihat_data');</code>
	<code>}</code>



Fungsi tambah data kecelakaan digunakan untuk menambahkan data kecelakaan pada setiap lokasi yang sudah disimpan di *database* sebelumnya. Fungsi ini dimulai dengan mengumpulkan data-data yang dibutuhkan. Data terlebih dahulu ditampung menggunakan beberapa *variable*, antara lain \$kode\_titik, \$titik, \$id\_jalan, \$nama\_jalan, \$kecamatan, \$sud\_wilayah, \$wilayah\_id, \$meninggal, \$luka\_berat, \$luka\_ringan, \$total, \$waktu, \$tipe\_geo, dan \$koordinat. Setelah itu data yang sudah ditampung akan digabungkan ke dalam bentuk *array* dengan nama \$data. Struktur data *array* disesuaikan dengan struktur *table* di *database* yang akan menyimpan data kecelakaan baru. Selanjutnya data *array* tersebut dikirimkan ke *model m\_kelola*. Data berhasil disimpan dan dapat ditampilkan pada halaman Kelola Data Kecelakaan.

### 5.2.3 Menampilkan Peta

Pada proses menampilkan peta, fungsi yang digunakan ada 4 yaitu *index*, *tampil\_peta*, *tampil\_wilayah*, dan *tampil\_jalan*. Keempat fungsi yang digunakan berada didalam *controller* peta. Semua fungsi tersebut harus dijalankan, jika tidak dijalankan salah satu maka akan terjadi *error* sehingga menyebabkan peta tidak dapat ditampilkan.

#### 5.2.3.1 Fungsi Index

Pada Tabel 5.8 merupakan *source code* dari fungsi *index*.

**Table 5.8 Source Code Fungsi Index**

1	<code>public function index()</code>
2	<code>{</code>
3	<code>    \$this-&gt;load-&gt;library('session');</code>
4	<code>    if (\$this-&gt;session-&gt;userdata('nama')== 'Theo') {</code>
5	<code>        \$data['apapun'] = '&lt;td&gt;&lt;ahref="'.site_url('c_kelola/ubah_data/').'\'+(feature.properties['\ID_KECELAKAAN\'] !=null?Autolinker.link(String (feature. properties['\ID_KECELAKAAN\'])): '\'\')+'\\"&gt;&lt;button type=\"button\"name=\"button \"/&gt;Ubah Data Kecelakaan&lt;/button&gt; &lt;/a&gt;&lt;/td&gt;\';</code>
6	<code>    }</code>
7	<code>    else {</code>
8	<code>        \$data['apapun'] = "";</code>
9	<code>        echo \$this-&gt;session-&gt;userdata('nama');</code>
10	<code>    }</code>
11	<code>    \$data['p'] = \$this-&gt;tampil_peta();</code>
12	<code>    \$data['w'] = \$this-&gt;tampil_wilayah();</code>
13	<code>    \$data['j'] = \$this-&gt;tampil_jalan();</code>
14	<code>    \$this-&gt;load-&gt;view('index',\$data);</code>
15	<code>}</code>

Fungsi *index* digunakan untuk mengambil komponen data atribut dan data spasial yang telah ditampung terlebih dahulu pada fungsi *tampil peta*, *tampil wilayah*, dan *tampil jalan*. Komponen-komponen yang telah diambil dikonversi

menjadi data *array*. Setelah itu data *array* tersebut dikirimkan ke *view index*. Pada fungsi ini terdapat juga sebuah tombol yang hanya bisa digunakan jika sudah berhasil *login* sebagai admin. Hasil dari fungsi ini dapat dilihat pada Gambar 5.12.

### 5.2.3.2 Fungsi Tampil Peta

Pada Tabel 5.9 merupakan *source code* dari fungsi tampil peta.

**Table 5.9 Source Code Fungsi Tampil Peta**

1	<code>public function tampil_peta()</code>
2	<code>{</code>
3	<code>    \$this-&gt;load-&gt;model('m_kelola');</code>
4	<code>    \$rekap_titik = \$this-&gt;m_kelola-&gt;rekap_titik()-&gt;result_array();</code>
5	<code>    \$file['type'] = "FeatureCollection";</code>
6	<code>    \$file['crs']['type'] = "name";</code>
7	<code>    \$file['crs']['properties']['name'] = "urn:ogc:def:crs:OGC:1.3:CRS84";</code>
8	<code>    foreach (\$rekap_titik as \$key =&gt; \$value) {</code>
9	<code>        \$file['features'][\$key]['type'] = "Feature";</code>
10	<code>        \$idk = \$rekap_titik[\$key]['ID_KECELAKAAN'];</code>
11	<code>        \$file['features'][\$key]['properties']['ID_KECELAKAAN'] = (double)number_format(\$idk, 1);</code>
12	<code>        \$file['features'][\$key]['properties']['KODE_TITIK'] = \$rekap_titik[\$key]['KODE_TITIK'];</code>
13	<code>        \$file['features'][\$key]['properties']['ID_JALAN'] = (float)\$rekap_titik[\$key]['ID_JALAN'];</code>
14	<code>        \$file['features'][\$key]['properties']['ID_WILAYAH'] = (float)\$rekap_titik[\$key]['ID_WILAYAH'];</code>
15	<code>        \$l = json_decode(\$rekap_titik[\$key]['KOORDINAT']);</code>
16	<code>        \$ln = \$l[0];</code>
17	<code>        \$lt = \$l[1];</code>
18	<code>        \$file['features'][\$key]['properties']['Longitude'] = \$ln;</code>
19	<code>        \$file['features'][\$key]['properties']['Latitude'] = \$lt;</code>
20	<code>        \$file['features'][\$key]['properties']['Nama_Jalan'] = \$rekap_titik[\$key]['Name'];</code>
21	<code>        \$file['features'][\$key]['properties']['Kecamatan'] = \$rekap_titik[\$key]['KECAMATAN'];</code>
22	<code>        \$file['features'][\$key]['properties']['Meninggal'] = (float)\$rekap_titik[\$key]['M'];</code>
23	<code>        \$file['features'][\$key]['properties']['Luka_Berat'] = (float)\$rekap_titik[\$key]['LB'];</code>
24	<code>        \$file['features'][\$key]['properties']['Luka_Ringan'] = (float)\$rekap_titik[\$key]['LR'];</code>
25	<code>        \$file['features'][\$key]['properties']['Total Kecelakaan'] = (float)\$rekap_titik[\$key]['T'];</code>

```

26     $file['features'][$key]['properties']['Waktu'] =
(float)$rekap_titik[$key]['WAKTU'];
27     $file['features'][$key]['geometry']['type'] =
$rekap_titik[$key]['TIPE_GEO'];
28     $file['features'][$key]['geometry']['coordinates']
= json_decode($rekap_titik[$key]['KOORDINAT']);
29     }
30     $data_json = json_encode($file);
31     return $data_json;
32 }

```

Fungsi Tampil Peta merupakan fungsi yang digunakan untuk membuat komponen dari peta dalam bentuk *json*. Sebelum data dikonversi menjadi *json*, diperlukan beberapa data dari *database*. Data tersebut diambil dengan memuat *model* *m\_kelola*. Kemudian data yang sudah didapatkan diubah menjadi *array* dan setelah itu dikonversi menjadi *json*.

### 5.2.3.3 Fungsi Tampil Wilayah

Pada Tabel 5.10 merupakan *source code* dari fungsi tampil wilayah.

**Table 5.10 Source Code Tampil Wilayah**

```

1  public function tampil_wilayah()
2  {
3      $this->load->model('m_kelola');
4      $wilayah = $this->m_kelola->lihat_wilayah()-
>result_array();
5      $file['type'] = "FeatureCollection";
6      $file['crs']['type'] = "name";
7      $file['crs']['properties']['name'] = "urn:ogc:def:crs:OGC:
1.3:CRS84";
8      foreach ($wilayah as $key => $value) {
9          $file['features'][$key]['type'] = "Feature";
10         $file['features'][$key]['properties']['ID'] =
$wilayah[$key]['ID_WILAYAH'];
11         $file['features'][$key]['properties']['Kecamatan']
= $wilayah[$key]['KECAMATAN'];
12         $file['features'][$key]['properties']['xcoord'] =
(double)$wilayah[$key]['XCOORD'];
13         $file['features'][$key]['properties']['ycoord'] =
(double)$wilayah[$key]['YCOORD'];
14         $file['features'][$key]['geometry']['type'] =
$wilayah[$key]['TIPE_GEO'];
15         $file['features'][$key]['geometry']['coordinates']
= "[".$wilayah[$key]['KOORDINAT']."";
16         $file['features'][$key]['geometry']['coordinates']
= json_decode($file['features'][$key]['geometry']['coordinates
']);
17     }

```

```

18     $data_json = json_encode($file);
19     return $data_json;
20 }

```

Fungsi Tampil Wilayah merupakan fungsi yang digunakan untuk membuat komponen dari peta dalam bentuk *json*. Sebelum data dikonversi menjadi *json*, diperlukan beberapa data dari *database*. Data tersebut diambil dengan memuat *model* *m\_kelola* yang kemudian disusun sesuai dengan urutannya. Kemudian data yang sudah didapatkan diubah menjadi *array* dan setelah itu dikonversi menjadi *json*.

#### 5.2.3.4 Fungsi Tampil Jalan

Pada Tabel 5.11 merupakan *source code* dari fungsi tampil jalan.

**Table 5.11 Source Code Tampil Jalan**

```

1  public function tampil_jalan()
2  {
3      $this->load->model('m_kelola');
4      $jalan = $this->m_kelola->jalan_total()->result_array();
5      $file['type'] = "FeatureCollection";
6      $file['crs']['type'] = "name";
7      $file['crs']['properties']['name'] = "urn:ogc:def:crs:OGC:
1.3:CRS84";
8      foreach ($jalan as $key => $value) {
9          $file['features'][$key]['type'] = "Feature";
10         $file['features'][$key]['properties']['ID'] =
11         $jalan[$key]['ID_JALAN'];
12         $file['features'][$key]['properties']['Type'] =
13         $jalan[$key]['TypeD'];
14         $file['features'][$key]['properties']['Category'] =
15         $jalan[$key]['Category'];
16         $file['features'][$key]['properties']['Name'] =
17         $jalan[$key]['Name'];
18         $file['features'][$key]['properties']['SHAPE_len']
19         = $jalan[$key]['SHAPE_len'];
20         $file['features'][$key]['properties']['Total_K'] =
21         $jalan[$key]['TOTAL_JALAN'];
22         $file['features'][$key]['geometry']['type'] =
23         $jalan[$key]['TIPE_GEO'];
24         $file['features'][$key]['geometry']['coordinates']
25         = json_decode($jalan[$key]['KOORDINAT']);
26     }
27     $data_json = json_encode($file);
28     return $data_json;
29 }

```

Fungsi Tampil Jalan merupakan fungsi yang digunakan untuk membuat komponen dari peta dalam bentuk *json*. Sebelum data dikonversi menjadi *json*,

diperlukan beberapa data dari *database*. Data tersebut diambil dengan memuat *model* *m\_kelola* yang kemudian disusun sesuai dengan urutannya. Kemudian data yang sudah didapatkan diubah menjadi *array* dan setelah itu dikonversi menjadi *json*.

## 5.2.4 Menampilkan Statistik

Pada Tabel 5.12 merupakan *source code* dari fungsi statistik.

**Table 5.12 Source Code Menampilkan Statistik**

1	<code>function h_statistik() {</code>
2	<code>    // statistik jalan sering kecelakaan</code>
3	<code>        \$data['hitung_jalan'] = \$this-&gt;m_jalan-&gt;get_hitung_jalan()-&gt;result_array();</code>
4	<code>    // statistik kecelakaan setiap tahun</code>
5	<code>        \$data['hitung_tahun'] = \$this-&gt;m_datakecelakaan-&gt;get_hitung_tahun()-&gt;result_array();</code>
6	<code>    // statistik kecelakaan wilayah</code>
7	<code>        \$total_k = \$this-&gt;m_wilayah-&gt;get_hitung_wilayah()-&gt;result_array();</code>
8	<code>        \$grafik_total_k = array();</code>
9	<code>        foreach (\$total_k as \$key =&gt; \$value) {</code>
10	<code>            \$grafik_total_k[\$key]['label'] = \$total_k[\$key]['KECAMATAN'];</code>
11	<code>            \$grafik_total_k[\$key]['value'] = (int)\$total_k[\$key]['TOTAL'];</code>
12	<code>        }</code>
13	<code>        \$data['grafik_total_k'] = \$grafik_total_k;</code>
14	<code>    // statistik korban berdasarkan bulan</code>
15	<code>        \$data['hitung_korban'] = \$this-&gt;m_datakecelakaan-&gt;get_hitung_korban()-&gt;result_array();</code>
16	<code>    //analisis ean</code>
17	<code>        \$hitung = \$this-&gt;m_jalan-&gt;analisis()-&gt;result_array();</code>
18	<code>        \$hasil = array();</code>
19	<code>        foreach (\$hitung as \$key=&gt;\$value){</code>
20	<code>            \$hasil[]=array('Name'=&gt;\$value['Name'], 'ANALISIS'=&gt;(\$value['M']*10)+(\$value['LB']*4.25)+(\$value['LR']*2.33)+(\$value['T']*1));</code>
21	<code>        }</code>
22	<code>        function sortByOrder(\$a, \$b) {</code>
23	<code>            if(\$a['ANALISIS']==\$b['ANALISIS']) return 0;</code>
24	<code>            return \$a['ANALISIS'] &lt; \$b['ANALISIS']?1:-1;</code>
25	<code>        }</code>
26	<code>        usort(\$hasil, 'sortByOrder');</code>
27	<code>        \$hasil = array_slice(\$hasil, 0, 5);</code>
28	<code>        \$data['analisis'] = \$hasil;</code>

```
29 //hasil statistik
30     $this->load->view('statistik',$data);
31 }
```

Fungsi Statistik merupakan fungsi yang digunakan untuk mengolah dan menampilkan beberapa analisis. Pada fungsi ini membutuhkan beberapa data dari *database* yang didapatkan dengan cara memuat beberapa *model*, antara lain *m\_jalan*, *mm\_datakecelakaan*, dan *m\_wilayah*. Terdapat beberapa data *array* yang menyimpan data-data statistik. Beberapa data *array* tersebut antara lain *\$data['hitung\_jalan']* yang menyimpan data statistik jalan yang memiliki total kecelakaan terbanyak, *\$data['hitung\_tahun']* yang menyimpan data statistik jumlah kecelakaan setiap tahun, *\$data['grafik\_total\_k']* yang menyimpan data statistik jumlah kecelakaan setiap wilayah, *\$data['hitung\_korban']* yang menyimpan data statistik jumlah korban berdasarkan bulan, dan *\$data['analisis']* yang menyimpan hasil perhitungan *EAN*. Perhitungan *EAN* dilakukan dimana pembobotan diberikan pada masing-masing kategori. Perhitungan *EAN* terdapat pada baris 17-28. Pembobotan diberikan pada *value M* dengan bobot 10, *value LB* dengan bobot 4,25, dan *value LR* dengan bobot 2,33.

### 5.3 Implementasi Antarmuka

Mengacu pada bab 4 mengenai perancangan antarmuka, dibawah ini merupakan halaman antarmuka dari sistem. Berikut adalah *screenshot* dari implementasi antarmuka yang telah dibuat.

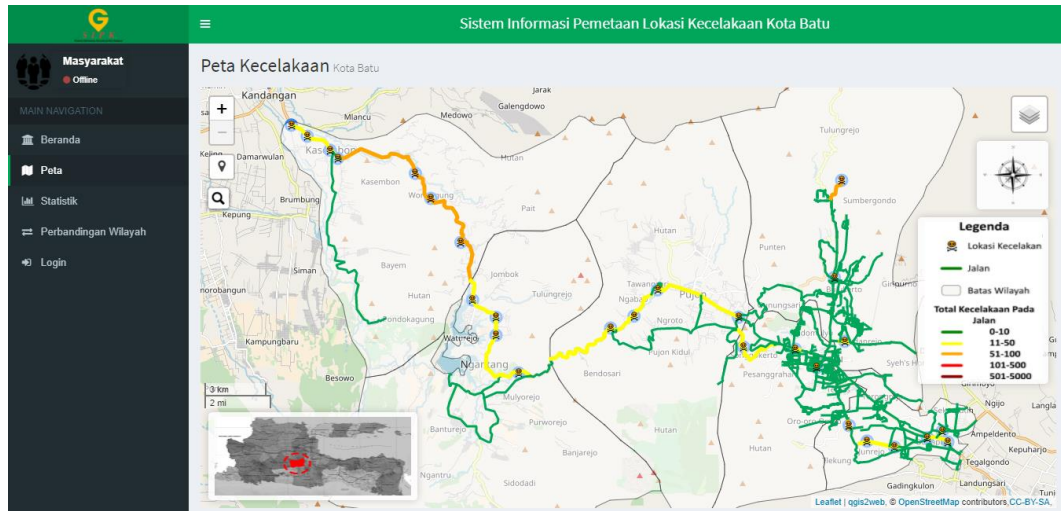
#### 5.3.1 Halaman Utama



Gambar 5.11 Antarmuka Halaman Utama

Gambar 5.11 merupakan halaman selamat datang di sistem. Halaman ini dapat diakses oleh masyarakat maupun admin. Pada halaman ini terdapat deskripsi tentang sistem yang telah dibuat.

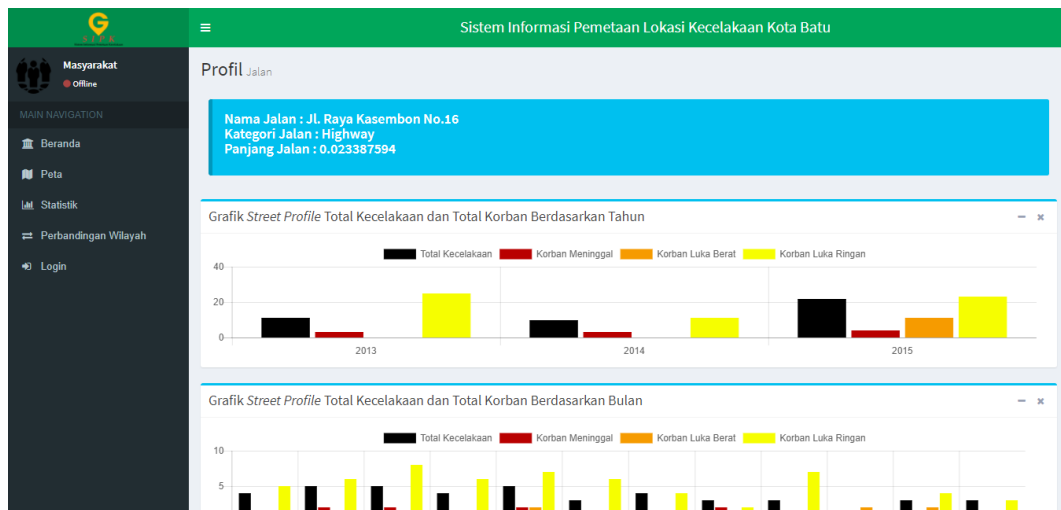
### 5.3.2 Halaman Peta



Gambar 5.12 Antarmuka Halaman Peta

Gambar 5.12 merupakan halaman peta saat masyarakat maupun admin mengakses menu peta. Pada halaman peta dapat ditunjukkan lokasi-lokasi kecelakaan yang pernah terjadi di Kota Batu dan sekitarnya beserta legenda. Terdapat juga peta jaringan jalan yang berwarna sesuai dengan jumlah kecelakaan yang terjadi pada setiap ruas jalan.

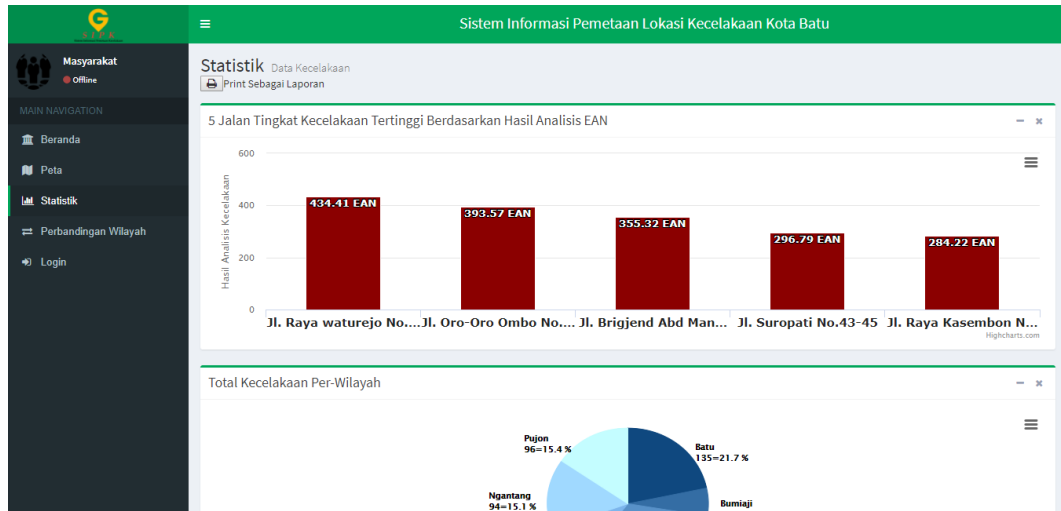
### 5.3.3 Halaman Detail Jalan



Gambar 5.13 Antarmuka Halaman Detail Jalan

Gambar 5.13 merupakan halaman detail jalan. Halaman ini dapat diakses oleh masyarakat maupun admin. Pada halaman ini ditunjukkan profil jalan yang berisi nama jalan, kategori jalan, dan panjang jalan. Terdapat juga grafik berbentuk bar yang menggambarkan total dan korban kecelakaan pada ruas jalan yang dipilih dalam 2 versi, yaitu per-bulan dan per-tahun.

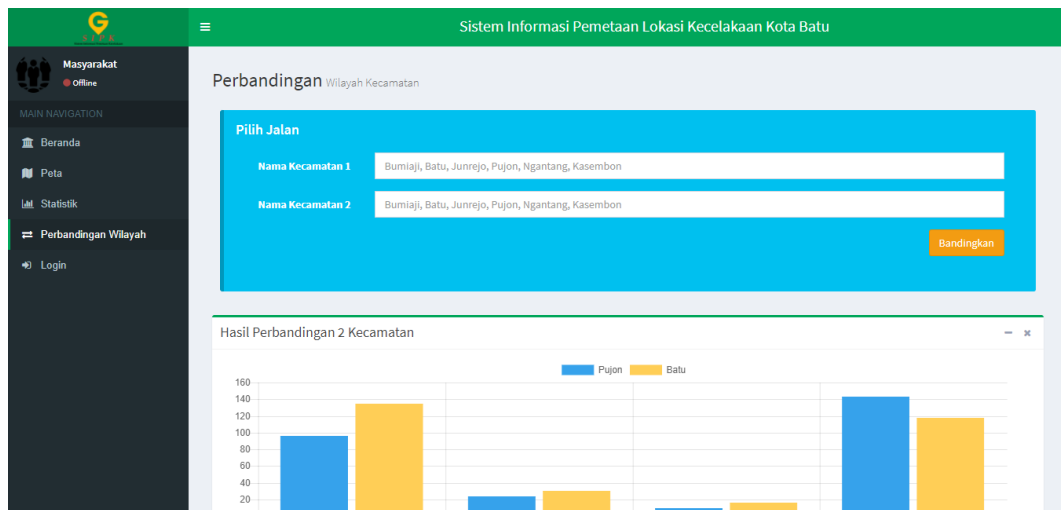
### 5.3.4 Halaman Statistik



Gambar 5.14 Antarmuka Halaman Statistik

Gambar 5.14 merupakan halaman statistik. Halaman ini dapat diakses oleh masyarakat maupun admin. Halaman menampilkan beberapa statistik kecelakaan dalam bentuk grafik. Terdapat grafik *donut* yang menggambarkan total kecelakaan per-kecamatan, grafik *line* yang menggambarkan total kecelakaan per-tahun, dan grafik *bar* yang menggambarkan semua total korban kecelakaan per-bulan.

### 5.3.5 Halaman Perbandingan Wilayah

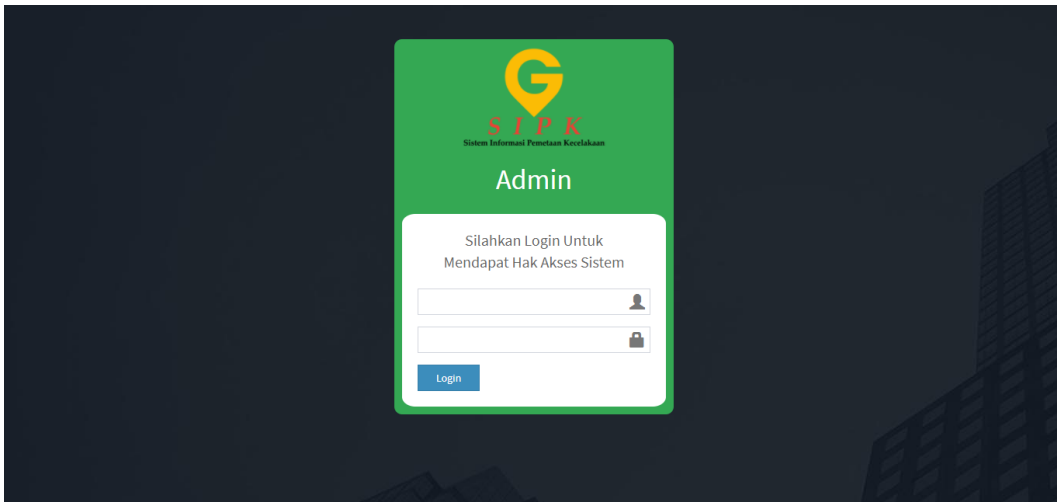


Gambar 5.15 Antarmuka Halaman Perbandingan Wilayah

Gambar 5.15 merupakan antarmuka halaman perbandingan wilayah. Halaman ini digunakan untuk melihat perbandingan antara 2 kecamatan. Informasi yang ditampilkan adalah perbandingan total kecelakaan, total korban meninggal, total korban luka berat, dan total korban luka ringan.



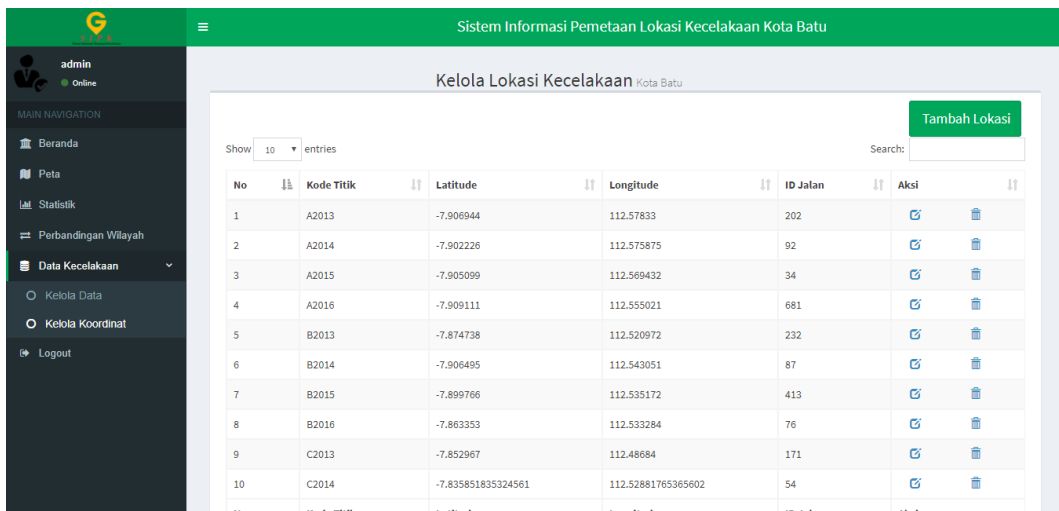
### 5.3.6 Halaman Login



**Gambar 5.16** Antarmuka Halaman Login

Gambar 5.16 merupakan halaman login yang digunakan untuk melakukan login sehingga dapat masuk ke halaman admin. Terdapat *field* untuk memasukkan *username* dan *password* beserta tombol *login*.

### 5.3.7 Halaman Kelola Lokasi



**Gambar 5.17** Antarmuka Halaman Kelola Lokasi

Gambar 5.17 merupakan halaman kelola lokasi yang hanya dapat diakses oleh admin. Data lokasi kecelakaan ditampilkan dalam bentuk tabel pada halaman ini. Tombol “Tambah Lokasi” pada halaman ini dapat digunakan untuk menghubungkan ke halaman tambah lokasi. Terdapat 5 kolom pada halaman ini, antara lain kolom “No” digunakan untuk menampung Nomor lokasi kecelakaan, kolom “Kode Titik” digunakan untuk menampung Kode Titik kecelakaan, kolom “Latitude” digunakan untuk menampung Latitude lokasi kecelakaan, kolom “Longitude” digunakan untuk menampung Longitude lokasi kecelakaan, dan

kolom “Aksi” untuk menampung operasi hapus atau ubah. Pada halaman ini admin dapat memilih fungsi ubah dan hapus untuk setiap data.

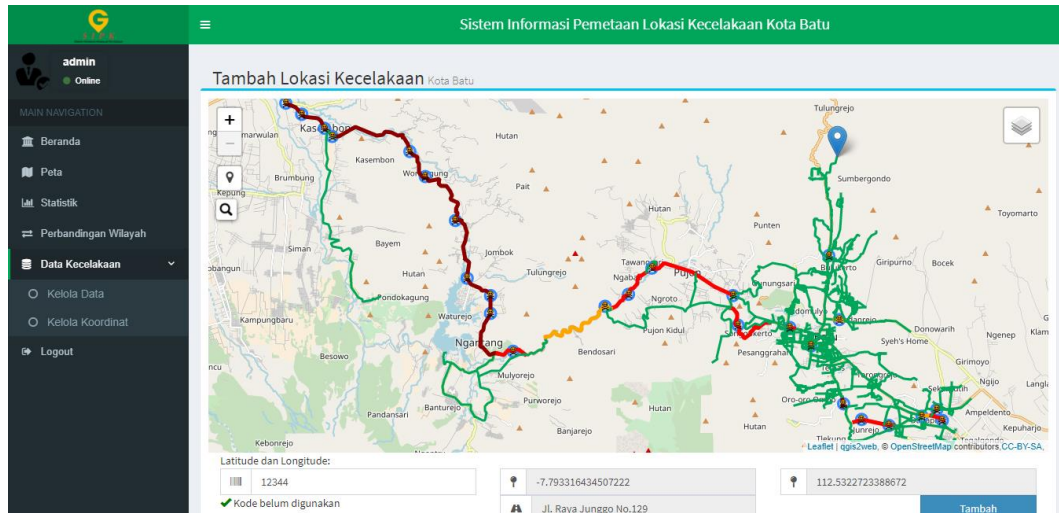
### 5.3.8 Halaman Kelola Data Kecelakaan

No	Nama Jalan	Kecamatan	Meninggal	Luka Berat	Luka Ringan	Total Kecelakaan	Waktu	Koordinat	Aksi
1	Jl. Raya Dadaprejo No.1	Junrejo	0	0	2	2	2013-02-01	[112.57833,-7.906944]	[Edit][Hapus]
2	Jl. Drs. Moh. Hatta No.4	Junrejo	1	0	1	3	2014-01-01	[112.575875,-7.902226]	[Edit][Hapus]
3	Jl. Raya Dadaprejo No.136	Junrejo	0	1	0	2	2015-01-01	[112.569432,-7.905099]	[Edit][Hapus]
4	Jl. Hasanuddin No.145	Junrejo	1	2	3	4	2016-01-01	[112.555021,-7.909111]	[Edit][Hapus]
5	Jl. Suropati No.43-45	Batu	1	0	5	6	2013-01-01	[112.520972,-7.874738]	[Edit][Hapus]
6	Jl. Brigjend Abd Manan Wijaya No. 1	Junrejo	1	0	4	3	2014-01-01	[112.543051,-7.906495]	[Edit][Hapus]
7	Jl. Oro-Oro Ombo No.200	Junrejo	1	2	1	4	2015-01-	[112.535172,-7.899766]	[Edit][Hapus]

**Gambar 5.18 Antarmuka Halaman Kelola Data Kecelakaan**

Gambar 5.18 merupakan halaman kelola data kecelakaan yang hanya dapat diakses oleh admin. Halaman ini menampilkan data kecelakaan dalam bentuk tabel. Terdapat 10 kolom pada halaman ini, antara lain kolom “No” digunakan sebagai *header* untuk nomor data kecelakaan, untuk kolom “Nama Jalan” digunakan sebagai *header* untuk data Nama Jalan, kolom “Kecamatan” untuk *header* data Kecamatan. Kemudian kolom “Meninggal”, “Luka Berat”, dan “Luka Ringan” sebagai *header* data korban kecelakaan. Selanjutnya kolom “Total Kecelakaan” digunakan sebagai *header* data total kecelakaan, kolom “Waktu” digunakan sebagai *header* data waktu kecelakaan, kolom Koordinat untuk menampung data koordinat lokasi kecelakaan, dan kolom “Aksi” untuk menampung operasi ubah dan hapus. Pada halaman ini admin dapat memilih fungsi ubah dan hapus untuk setiap data.

### 5.3.9 Halaman Tambah Lokasi Kecelakaan



Gambar 5.19 Antarmuka Halaman Tambah Lokasi Kecelakaan

Gambar 5.19 merupakan halaman tambah lokasi kecelakaan yang hanya dapat diakses oleh admin. Pada halaman ini admin dapat memilih lokasi baru untuk ditambahkan dalam daftar lokasi kecelakaan.

### 5.3.10 Halaman Tambah Data Kecelakaan



Gambar 5.20 Antarmuka Halaman Tambah Data Kecelakaan

Gambar 5.20 merupakan halaman tambah data kecelakaan yang hanya dapat diakses oleh admin. Pada halaman ini admin dapat menambahkan data kecelakaan pada lokasi tertentu. Terdapat beberapa *field* yang dapat diisi oleh admin, antara lain *field* Kode Titik, Nama Jalan, Kecamatan, Total, Meninggal, Luka Berat, Luka Ringan, dan Waktu.