Projek Kecil IB
Spatial Database

IF4040 PEMODELAN DATA LANJUT

Trapsilo Pramudya Bumi / 13510052
Hanif Eridaputra / 13510091

DESKRIPSI PERSOALAN

Angkutan Kota (Angkot), merupakan salah satu transportasi yang paling banyak dan paling sering digunakan oleh orang Indonesia salah satunya adalah di Bandung. Setiap angkot di Bandung tentunya memiliki jenis dan rute masing-masing sehingga untuk mencapai suatu tempat tertentu terkadang harus menaiki dua atau lebih angkot. Permasalahannya adalah angkot di Bandung cukup banyak dan tidak semua orang mengetahui rute yang dilewati oleh angkot tersebut. Keterangan rute yang tertera pada angkot biasanya hanya menuliskan terminal awal dan akhir yang dilewati angkot tersebut. Maka dari itu, dibutuhkan sebuah aplikasi yang dapat menyimpan data-data spatial dari rute angkot yang ada di bandung sehingga masyarakat bandung dapat mencari rute untuk mencapai suatu tempat tertentu dengan menggunakan angkot.

TEKNOLOGI YANG DIGUNAKAN

1. **PostgreSQL**
   Salah satu basis data relasional yang biasa digunakan.
2. **PostGIS**
   Estensi dari PostgreSQL yang mengurusi masalah basis data spatial dan query-query spatial.
3. **Google Maps Javascript API v3**
   Tampilan map dari Google.
4. **PHP**
   Bahasa standar yang biasa digunakan untuk aplikasi berbasis web.

DESAIN UMUM APLIKASI

Aplikasi yang dibuat adalah aplikasi berbasis Web yang memanfaatkan bahasa PHP dan basis data yang dikelola menggunakan PostgreSQL. Aplikasi ini akan memiliki beberapa fitur utama yaitu sebagai berikut:

1. User dapat menemukan angkot apa saja yang harus digunakan untuk mencapai suatu tujuan tertentu dengan menunjuk tempat asal dan tempat tujuan pada peta yang telah ditentukan.
2. User dapat melihat peta rute dari angkot yang akan digunakan.
3. User dapat mengetahui jarak tempuh dari tempat asal ke tempat tujuan.

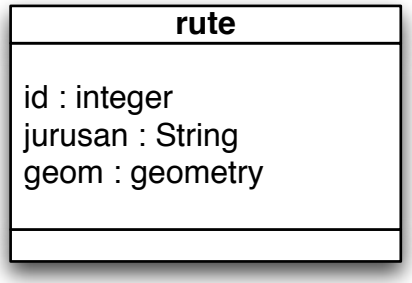
RUTE ANGKOT BANDUNG

Data sekitar 38 jenis angkot yang ada, hanya sebanyak 16 jenis yang dimasukkan ke dalam basis data yaitu sebagai berikut:

1. ABDUL MUIS – CICAHEUM VIA BINONG
2. ABDUL MUIS – DAGO
3. ABDUL MUIS – LEDENG
4. CICAHEUM – LEDENG
5. CICAHEUM – CIROYOM
6. CICAHEUM – CIBADUYUT
7. STASIUN HALL – DAGO
8. STASIUN HALL – CIUMBEULEUIT VIA CIHAMPELAS
9. STASIUN HALL – SARIJADI
10. STASIUN HALL – GUNUNG BATU
11. MARGAHAYU RAYA – LEDENG
12. CIROYOM – SARIJADI VIA SUKAJADI
13. SEDERHANA – CIJERAH
14. CIWASTRA – UJUNG BERUNG
15. ABDUL MUIS – MENGGER
16. CIBOGO ATAS – HALTEU ANDIR

SKEMA BASIS DATA

Basis data yang digunakan sangat sederhana dan hanya memiliki satu buah tabel seperti gambar di bawah. Atribut geom merupakan atribut basis data spatial yang akan menyimpan data spatial dalam bentuk garis rute angkot.



QUERY POSTGIS YANG DIGUNAKAN

`geometry ST_GeomFromText(text WKT)`
Mengembalikan bentuk geometry dari masukan teks dalam bentuk Well-Known Text (WKT)
Implementasi: Untuk membuat geometry dari masukan koordinat, digunakan query

```
SELECT ST_GeomFromText('POINT($koordinat_asal)')
```

dimana `$koordinat_asal` adalah koordinat bujur-lintang dalam bentuk desimal

`text ST_AsGeoJSON(geometry A)`
Mengembalikan bentuk GeoJSON dari masukan geometri
Implementasi: Untuk mengembalikan GeoJSON sebagai hasil kalkulasi rute angkot, digunakan query

```
SELECT ST_AsGeoJSON('$rute_akhir')
```

dimana `$rute_akhir` adalah geometry yang akan dikembalikan dalam bentuk GeoJSON.

`boolean ST_Intersects(geometry A, geometry B)`
Mengembalikan nilai boolean true jika geometry A berpotongan dengan geometry B
Implementasi: Untuk mengembalikan seluruh rute angkot yang berpotongan dengan titik awal digunakan query

```
SELECT geom FROM rute WHERE ST_Intersects('$titik_asal', geom)
```

dimana `$titik_asal` adalah geometry Point dan `geom` adalah geometry rute angkot

`boolean ST_Overlaps(geometry A, geometry B)`
Mengembalikan nilai boolean true jika geometry A memiliki sebagian ruang yang sama dengan geometry B, berdimensi sama, tetapi tidak ada satu geometry yang melingkupi yang lain
Implementasi: Untuk mencari jurusan angkot yang melingkupi suatu garis, digunakan query

```
SELECT jurusan FROM rute WHERE ST_Overlaps(geom,
'$bagian_rute_akhir')
```

dimana `jurusan` adalah nama jurusan angkot, geom adalah geometry rute angkot, dan `$bagian_rute_akhir` adalah garis hasil kalkulasi sementara rute angkot yang harus digunakan

`boolean ST_Equals(geometry A, geometry B)`
Mengembalikan nilai boolean true jika geometry A mewakili ruang yang sama dengan geometry B
Implementasi: Untuk mencari rute lain yang berpotongan dengan suatu rute angkot, namun tidak sama dengan rute angkot itu sendiri digunakan query

```
SELECT geom FROM rute WHERE ST_Intersects('$rute_proses',
geom) and not ST_Equals('$rute_proses', geom)
```

dimana `geom` adalah geometry rute angkot dan $rute_proses adalah rute angkot yang sedang diproses

`geometry ST_Buffer(geometry A, float radius)`
Mengembalikan geometry ruang di sekeliling geometry A sejauh radius
Implementasi: Untuk membentuk lingkaran dari sebuah titik dengan radius sebesar 0,001 digunakan query

```
ST_Buffer('$titik_asal', 0,001)
```

Dengan `$titik_asal` adalah titik asal naik angkot

`geometry ST_LineSubstring(geometry a_linestring, float start, float end)`
Mengembalikan geometry yang merupakan bagian dari LineString `a_linestring`, dengan titik awal start dan titik akhir end yang merupakan pecahan dari total panjang A

Implementasi: Untuk menyeleksi garis setelah pengguna naik angkot digunakan query

```
SELECT ST_LineSubstring('$inter_geom', $location_point_awal,
1)
```

dimana `$inter_geom` adalah rute angkot yang berpotongan dengan titik awal, `$location_point_awal` adalah angka desimal yang menunjukkan lokasi titik awal relatif terhadap panjang rute

`geometry ST_Intersection(geometry A, geometry B)`
Mengembalikan geometry perpotongan geometry A dan geometry B
Implementasi: Untuk menyeleksi perpotongan antara dua rute angkot, digunakan query

```
SELECT ST_Intersection('$rute_akhir', '$rute_awal')
```

dimana `$rute_akhir` dan `$rute_awal` adalah geometry rute angkot

`geometry ST_StartPoint(geometry a_linestring)`
mengembalikan Point yang merupakan titik paling awal dari sebuah LineString `a_linestring`
Implementasi: Untuk menyeleksi titik awal sebuah garis rute angkot, digunakan query

```
SELECT ST_StartPoint('$lokasi_potong')
```

dimana `$lokasi_potong` adalah sebuah garis perpotongan rute angkot

`float ST_LineLocatePoint(geometry a_linestring, geometry a_point)`
Mengembalikan angka desimal antara 0 dan 1 yang merupakan lokasi titik terdekat pada LineString `a_linestring` dengan Point `a_point`
Implementasi: untuk mengembalikan lokasi perpotongan titik dalam suatu rute angkot digunakan query

```
SELECT ST_LineLocatePoint('$acuan', '$titik_potong')
```

dimana `$acuan` adalah garis rute angkot yang dijadikan acuan dan `$titik_potong` adalah titik perpotongan rute angkot dengan rute lainnya

`text ST_GeometryType(geometry A)`
Mengembalikan tipe geometry A
Implementasi:

```
SELECT ST_GeometryType('$titik_potong_geom')
```

`integer ST_NumGeometries(geometry A)`
Mengembalikan jumlah geometry yang terkandung dalam geometry A jika A adalah koleksi geometry, 1 jika A adalah geometry tunggal
Implementasi:

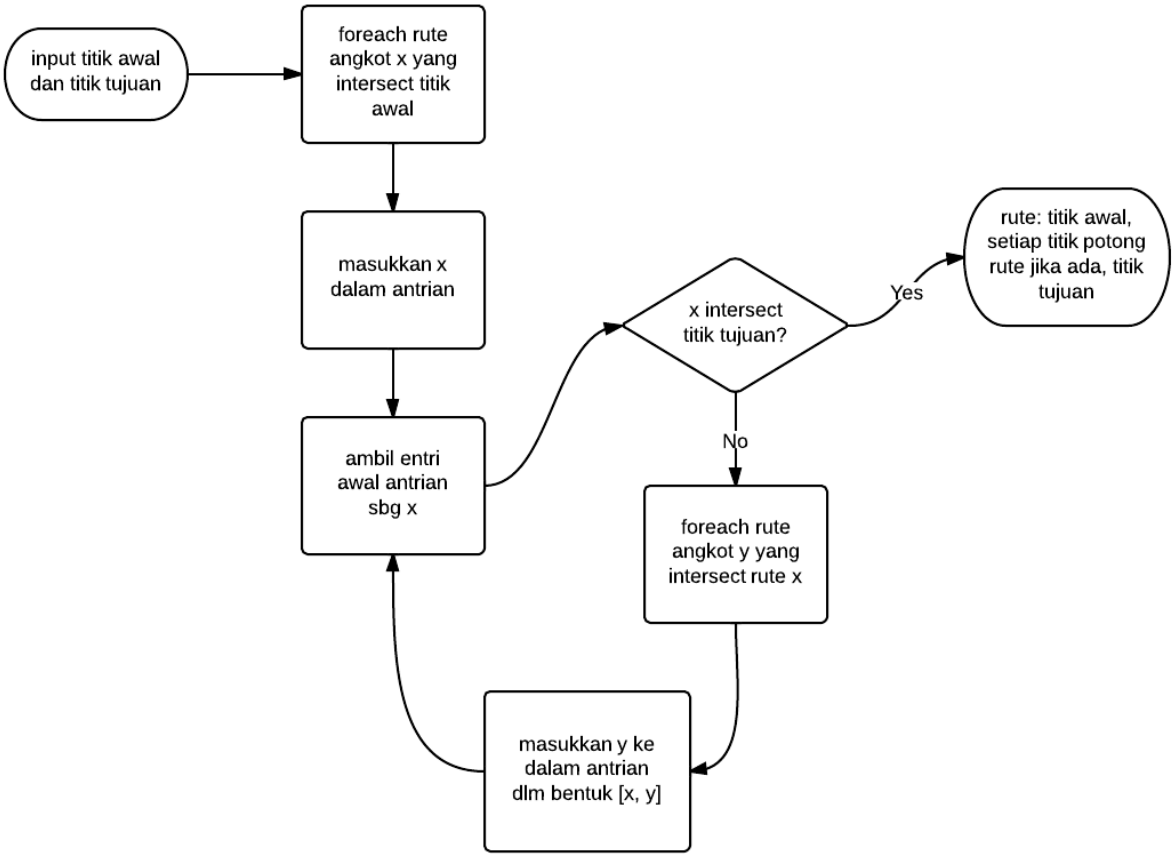
```
SELECT ST_NumGeometries('$titik_potong_geom')
```

`geometry ST_GeometryN(geometry A, integer N)`
Mengembalikan geometry ke-N jika A adalah koleksi geometry
Implementasi:

```
SELECT ST_GeometryN('$titik_potong_geom', $i)
```

ALGORITMA RUTE ANGKOT

Algoritma pencarian rute angkot adalah seperti gambar berikut:

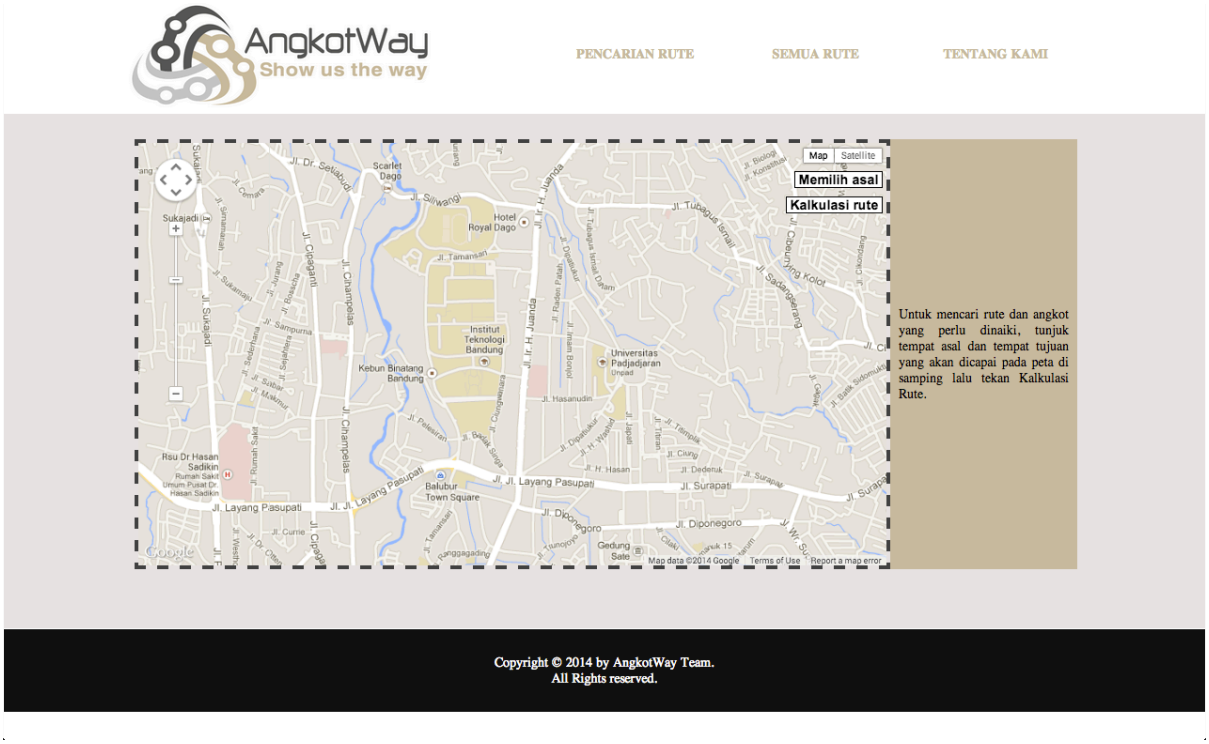


LANGKAH-LANGKAH MENJALANKAN APLIKASI

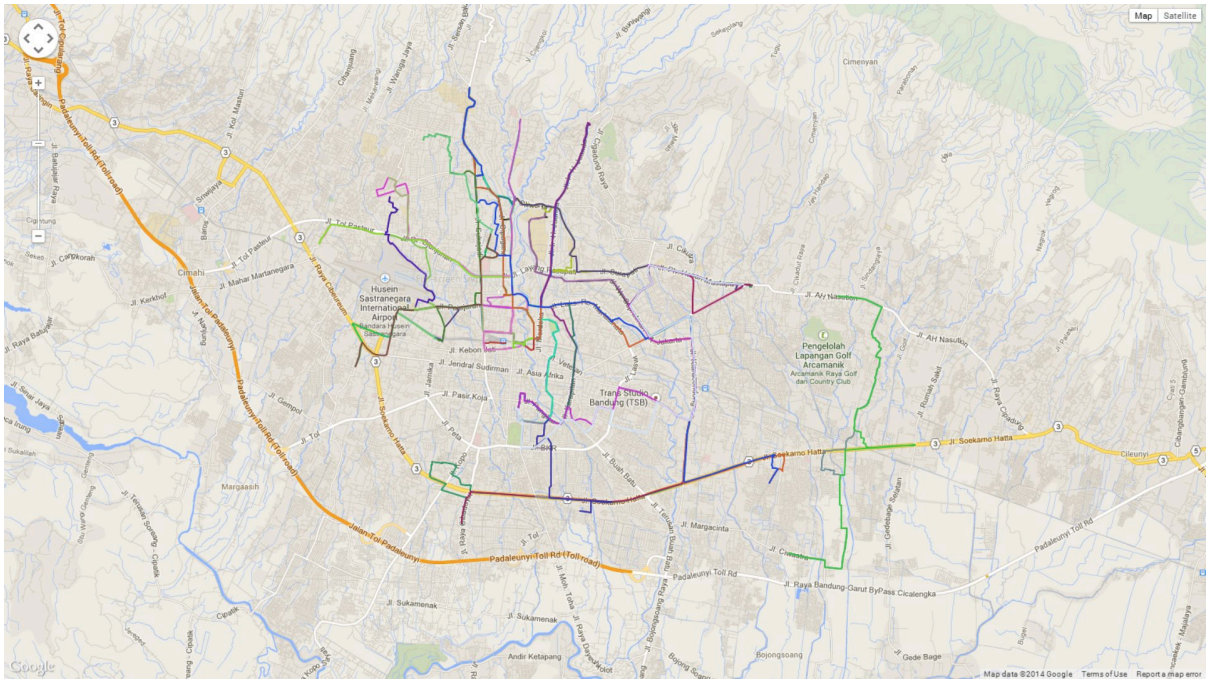
1. Pengguna memasukkan koordinat asal dan koordinat tujuan ke peta
2. Aplikasi(php) melakukan query ke postgre/postgis
3. Postgis mengembalikan jurusan angkot yang harus ditempuh
4. Peta menampilkan jurusan angkot yang dikembalikan

SCREENSHOT APLIKASI ANGKOTWAY

Antar muka halaman pencarian rute angkot:



Antar muka halaman semua rute:



Antar muka halaman tentang kami:

KESIMPULAN DAN SARAN

Kesimpulan dan insight yang didapatkan dari hasil eksplorasi PostGIS ini adalah sebagai berikut:

1. PostGIS merupakan basis data yang sudah sangat baik dalam pengelolaan basis data spatial.
2. Query dapat dijalankan dengan baik dalam PostGIS.
3. Aplikasi berbasis web dapat disambungkan dengan mudah dengan PostgreSQL dengan menggunakan PHP.

Saran untuk pengguna PostGIS, terdapat berbagai macam tools yang dapat digunakan untuk menggunakan PostGIS seperti QGIS, OpenJump, uDig dan lain-lain. Untuk membuat antarmuka juga dapat memanfaatkan MapServer, GeoServer dan lain-lain. Pilih tools yang paling cocok untuk membuat suatu aplikasi spatial dan jangan hanya terpaku pada satu tools.

REFERENSI

- http://gilangdawous.wordpress.com/2012/03/11/rute-angkot-bandung/
- http://postgis.net/docs/manual-2.1/reference.html
- https://developers.google.com/maps/documentation/javascript/3.exp/reference
- http://www.php.net/manual/en/ref.pgsql.php
- http://www.postgresql.org/docs/9.3/static/index.html

LAMPIRAN

query.php (semua query dan fungsi dimasukkan ke dalam satu file).

```
<?php
define('TOLERANCE', 0.001);

// koneksi ke PostgreSQL
$dbconn = pg_connect("host=localhost dbname=rute_angkot user=postgres
password=root") or die("Could not connect.");

$koordinat_asal = $_GET["titik_asal"];
$koordinat_tujuan = $_GET["titik_tujuan"];

$titik_asal = pg_fetch_result(pg_query($dbconn, "SELECT
ST_GeomFromText('POINT($koordinat_asal)')"), 0);
$titik_tujuan = pg_fetch_result(pg_query($dbconn, "SELECT
ST_GeomFromText('POINT($koordinat_tujuan)')"), 0);

$rute_antrian = array();
$intersection = pg_fetch_all(pg_query($dbconn, sprintf("SELECT geom FROM rute WHERE
ST_Intersects(ST_Buffer('$titik_asal', %f), geom)", TOLERANCE)));

foreach ($intersection as $inter) {
	$inter_geom = $inter["geom"];
	$location_point_awal = pg_fetch_result(pg_query($dbconn, "SELECT
ST_LineLocatePoint('$inter_geom', '$titik_asal')"), 0);
	$inter_substring = pg_fetch_result(pg_query($dbconn, "SELECT
ST_LineSubstring('$inter_geom', $location_point_awal, 1)"), 0);
	$rute_antrian[] = [$inter_substring];
}

while (count($rute_antrian) > 0) {
	$array_rute_proses = array_shift($rute_antrian);
	$rute_proses = $array_rute_proses[count($array_rute_proses)-1];
	$sudah_sampai_tujuan = pg_fetch_result(pg_query($dbconn, sprintf("SELECT
ST_Intersects('$rute_proses', ST_Buffer('$titik_tujuan', %f))", TOLERANCE)), 0);

	if ($sudah_sampai_tujuan == 't') {
		$all_routes = array();
		for ($i=0; $i < count($array_rute_proses); $i++) {
			$bagian_rute_akhir = $array_rute_proses[$i];
			if ($i == 0) {
				$location_point_awal = pg_fetch_result(pg_query($dbconn,
"SELECT ST_LineLocatePoint('$bagian_rute_akhir', '$titik_asal')"), 0);
			} else {
				$rute_sebelumnya = $array_rute_proses[$i-1];
				$location_point_awal =
cari_lokasi_titik_potong($rute_sebelumnya, $bagian_rute_akhir, $bagian_rute_akhir);
			}
			if ($i == count($array_rute_proses)-1) {
				$location_point_akhir =
pg_fetch_result(pg_query($dbconn, "SELECT ST_LineLocatePoint('$bagian_rute_akhir',
'$titik_tujuan')"), 0);
			} else {
				$rute_sesudahnya = $array_rute_proses[$i+1];
				$location_point_akhir =
cari_lokasi_titik_potong($bagian_rute_akhir, $rute_sesudahnya, $bagian_rute_akhir);
			}

			$rute_substring = pg_fetch_result(pg_query($dbconn, "SELECT
ST_AsGeoJSON(ST_LineSubstring('$bagian_rute_akhir', $location_point_awal,
$location_point_akhir))"), 0);
```

```
                        $decode_routes = json_decode($rute_substring, true);
                        $decode_routes["jurusan"] = pg_fetch_result(pg_query($dbconn,
"SELECT jurusan FROM rute WHERE ST_Overlaps(geom, '$bagian_rute_akhir')"), 0);
                        $all_routes[] = $decode_routes;
                }

                $json_routes = json_encode($all_routes);

                echo $json_routes;
                break;
        } else {
                $rute_lain_yg_intersect = pg_fetch_all(pg_query($dbconn, "SELECT geom
FROM rute WHERE ST_Intersects('$rute_proses', geom) and not
ST_Equals('$rute_proses', geom)"));
                foreach ($rute_lain_yg_intersect as $rt) {
                        $rt_geom = $rt["geom"];

                        $location_point_titik_potong =
cari_lokasi_titik_potong($rute_proses, $rt_geom, $rt_geom);

                        $inter_substring = pg_fetch_result(pg_query($dbconn, "SELECT
ST_LineSubstring('$rt_geom', $location_point_titik_potong, 1)"), 0);

                        array_push($array_rute_proses, $inter_substring);
                        $rute_antrian[] = $array_rute_proses;
                        array_pop($array_rute_proses);
                }
        }
}

function cari_lokasi_titik_potong($rute_awal, $rute_akhir, $acuan) {
        global $dbconn;

        $titik_potong_geom = pg_fetch_result(pg_query($dbconn, "SELECT
ST_Intersection('$rute_akhir', '$rute_awal')"), 0);
        $type = pg_fetch_result(pg_query($dbconn, "SELECT
ST_GeometryType('$titik_potong_geom')"), 0);

        if ($type == "ST_GeometryCollection" || $type == "ST_MultiPoint") {
                $num_geoms = pg_fetch_result(pg_query($dbconn, "SELECT
ST_NumGeometries('$titik_potong_geom')"), 0);
                for ($i=1; $i <= $num_geoms; $i++) {
                        $lokasi_potong = pg_fetch_result(pg_query($dbconn, "SELECT
ST_GeometryN('$titik_potong_geom', $i)"), 0);
                        if (pg_fetch_result(pg_query($dbconn, "SELECT
ST_GeometryType('$lokasi_potong')"), 0) == "ST_Point") {
                                $location_point_calon_titik_potong =
pg_fetch_result(pg_query($dbconn, "SELECT ST_LineLocatePoint('$acuan',
'$lokasi_potong')"), 0);
                        } else {
                                $lokasi_potong_awal = pg_fetch_result(pg_query($dbconn,
"SELECT ST_StartPoint('$lokasi_potong')"), 0);
                                $location_point_calon_titik_potong =
pg_fetch_result(pg_query($dbconn, "SELECT ST_LineLocatePoint('$acuan',
'$lokasi_potong_awal')"), 0);
                        }
                        if (!isset($location_point_titik_potong) ||
(int)$location_point_calon_titik_potong < (int)$location_point_titik_potong) {
                                $location_point_titik_potong =
$location_point_calon_titik_potong;
                        }
                }
        } else if ($type == "ST_MultiLineString") {
                $num_geoms = pg_fetch_result(pg_query($dbconn, "SELECT
ST_NumGeometries('$titik_potong_geom')"), 0);
                for ($i=1; $i <= $num_geoms; $i++) {
                        $line = pg_fetch_result(pg_query($dbconn, "SELECT
ST_GeometryN('$titik_potong_geom', $i)"), 0);
```

```
                    $calon_titik_potong = pg_fetch_result(pg_query($dbconn, "SELECT
ST_StartPoint('$line')"), 0);
                    $location_point_calon_titik_potong =
pg_fetch_result(pg_query($dbconn, "SELECT ST_LineLocatePoint('$acuan',
'$calon_titik_potong')"), 0);
                    if (!isset($location_point_titik_potong) ||
(int)$location_point_calon_titik_potong < (int)$location_point_titik_potong) {
                        $location_point_titik_potong =
$location_point_calon_titik_potong;
                    }
                }
        } else if ($type == "ST_LineString") {
                $titik_potong = pg_fetch_result(pg_query($dbconn, "SELECT
ST_StartPoint('$titik_potong_geom')"), 0);
                $location_point_titik_potong = pg_fetch_result(pg_query($dbconn,
"SELECT ST_LineLocatePoint('$acuan', '$titik_potong')"), 0);
        } else if ($type == "ST_Point") {
                $location_point_titik_potong = pg_fetch_result(pg_query($dbconn,
"SELECT ST_LineLocatePoint('$acuan', '$titik_potong_geom')"), 0);
        } else {
                var_dump($type);
                die("Unknown intersection type.");
        }
        return $location_point_titik_potong;
}

pg_close($dbconn);
?>
```

peta.html (file html utama yang digunakan untuk mencari rute)

```
<!DOCTYPE html>
<html>
  <head>
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no" />
    <style type="text/css">
      html { height: 100% }
      body { height: 100%; margin: 0; padding: 0 }
      #map-canvas { height: 100% }
    </style>
    <script type="text/javascript"
      src="http://maps.googleapis.com/maps/api/js?sensor=false">
    </script>
    <script type="text/javascript">
      var asal = null;
      var tujuan = null;
      var lines = [];

      function getRandomColor() {
        var letters = '0123456789ABCDEF'.split('');
        var color = '#';
        for (var i = 0; i < 6; i++ ) {
          color += letters[Math.round(Math.random() * 15)];
        }
        return color;
      }

      function emptyLines() {
        for (var i = 0; i < lines.length; i++) {
          lines[i].setMap(null);
        };
        lines = [];
      }

      PilihAsalTujuanControl.prototype.toggleAsal = true;
```

```
      function PilihAsalTujuanControl(controlDiv, map) {
        var control = this;

        // Set CSS styles for the DIV containing the control
        // Setting padding to 5 px will offset the control
        // from the edge of the map
        controlDiv.style.padding = '5px';

        // Set CSS for the control border
        var controlUI = document.createElement('div');
        controlUI.style.backgroundColor = 'white';
        controlUI.style.borderStyle = 'solid';
        controlUI.style.borderWidth = '1px';
        controlUI.style.cursor = 'pointer';
        controlUI.style.textAlign = 'center';
        controlUI.title = 'Klik untuk berubah antara memilih titik asal dan titik
tujuan';
        controlDiv.appendChild(controlUI);

        // Set CSS for the control interior
        var controlText = document.createElement('div');
        controlText.style.fontFamily = 'Arial,sans-serif';
        controlText.style.fontSize = '16px';
        controlText.style.paddingLeft = '4px';
        controlText.style.paddingRight = '4px';
        controlText.innerHTML = '<b>Memilih asal</b>';
        controlUI.appendChild(controlText);

        google.maps.event.addDomListener(controlUI, 'click', function() {
          if (!control.toggleAsal) {
            control.toggleAsal = true;
            controlText.innerHTML = '<b>Memilih asal</b>';
          }
          else {
            control.toggleAsal = false;
            controlText.innerHTML = '<b>Memilih tujuan</b>';
          }
        });

        google.maps.event.addListener(map, 'click', function(e) {
          if (control.toggleAsal) {
            if (asal != null) {
              asal.setMap(null);
              asal = null;
            }
            asal = new google.maps.Marker({
              position: e.latLng,
              map: map,
              animation: google.maps.Animation.DROP,
            });
            console.log("Asal: " + asal.getPosition().lng() + " " +
asal.getPosition().lat());
          } else {
            if (tujuan != null) {
              tujuan.setMap(null);
              tujuan = null;
            }
            tujuan = new google.maps.Marker({
              position: e.latLng,
              map: map,
              animation: google.maps.Animation.DROP,
            });
            console.log("Tujuan: " + tujuan.getPosition().lng() + " " +
tujuan.getPosition().lat());
          }
        })
      }
```

```
      function CalculateRouteControl(controlDiv, map) {

        // Set CSS styles for the DIV containing the control
        // Setting padding to 5 px will offset the control
        // from the edge of the map
        controlDiv.style.padding = '5px';

        // Set CSS for the control border
        var controlUI = document.createElement('div');
        controlUI.style.backgroundColor = 'white';
        controlUI.style.borderStyle = 'solid';
        controlUI.style.borderWidth = '1px';
        controlUI.style.cursor = 'pointer';
        controlUI.style.textAlign = 'center';
        controlUI.title = 'Klik untuk memulai kalkulasi rute';
        controlDiv.appendChild(controlUI);

        // Set CSS for the control interior
        var controlText = document.createElement('div');
        controlText.style.fontFamily = 'Arial,sans-serif';
        controlText.style.fontSize = '16px';
        controlText.style.paddingLeft = '4px';
        controlText.style.paddingRight = '4px';
        controlText.innerHTML = '<b>Kalkulasi rute</b>';
        controlUI.appendChild(controlText);

        google.maps.event.addDomListener(controlUI, 'click', function() {
            var xmlhttp = new XMLHttpRequest();
            xmlhttp.onreadystatechange = function() {
              if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
                console.log(xmlhttp.responseText);
                if (xmlhttp.responseText.indexOf("<") != -1) {
                  alert("error.");
                  return;
                }
                emptyLines();
                var jsonstring = eval(xmlhttp.responseText);
                var jurusan = "";
                for (var h = 0; h < jsonstring.length; h++) {
                  var lineStringCoords = [];
                  var GeoJSON = jsonstring[h];
                  for (var i = 0; i < GeoJSON.coordinates.length; i++) {
                    lineStringCoords.push(new
google.maps.LatLng(GeoJSON.coordinates[i][1], GeoJSON.coordinates[i][0]));
                  };
                  var lineString = new google.maps.Polyline({
                    path: lineStringCoords,
                    map: map,
                    strokeColor: getRandomColor(),
                    strokeOpacity: 1.0,
                    strokeWeight: 2
                  });
                  lines.push(lineString);
                  jurusan += GeoJSON.jurusan + " -> ";
                }
                jurusan = jurusan.substr(0, jurusan.length-4);
                alert("Jurusan yang harus dinaiki: " + jurusan);
              }
            };
            var querystring = "query.php?titik_asal="+asal.getPosition().lng() +
"%20" + asal.getPosition().lat()
              +"&titik_tujuan="+tujuan.getPosition().lng() + "%20" +
tujuan.getPosition().lat();
            console.log(querystring);
            xmlhttp.open("GET", querystring, true);
            xmlhttp.send();
        });
      }
```

```
      function initialize() {
        var itebe = new google.maps.LatLng(-6.892578,107.610397);
        var mapOptions = {
          center: itebe,
          zoom: 15,
          streetViewControl: false,
        };
        var map = new google.maps.Map(document.getElementById("map-canvas"),
            mapOptions);

        var pilihAsalTujuanControlDiv = document.createElement('div');
        var pilihAsalTujuanControl = new
PilihAsalTujuanControl(pilihAsalTujuanControlDiv, map);
        pilihAsalTujuanControlDiv.index = 1;

map.controls[google.maps.ControlPosition.RIGHT_TOP].push(pilihAsalTujuanControlDiv)
;

        var calculateRouteDiv = document.createElement('div');
        var calculateRouteControl = new CalculateRouteControl(calculateRouteDiv,
map);
        calculateRouteDiv.index = 1;

map.controls[google.maps.ControlPosition.RIGHT_TOP].push(calculateRouteDiv);
      }

      google.maps.event.addDomListener(window, 'load', initialize);
    </script>
  </head>
  <body>
    <div id="map-canvas" />
  </body>
</html>
```

showallroutes.html (file html yang digunakan untuk menampilkan semua rute)

```
<!DOCTYPE html>
<html>
  <head>
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no" />
    <style type="text/css">
      html { height: 100% }
      body { height: 100%; margin: 0; padding: 0 }
      #map-canvas { height: 100% }
    </style>
    <script type="text/javascript"
      src="http://maps.googleapis.com/maps/api/js?sensor=false">
    </script>
    <script type="text/javascript">
      var asal = null;
      var tujuan = null;

      function getRandomColor() {
        var letters = '0123456789ABCDEF'.split('');
        var color = '#';
        for (var i = 0; i < 6; i++ ) {
          color += letters[Math.round(Math.random() * 15)];
        }
        return color;
      }

      function initialize() {
        var bandung = new google.maps.LatLng(-
6.908704023900703,107.61880874633789);
```

```
        var mapOptions = {
          center: bandung,
          zoom: 13,
          streetViewControl: false,
        };
        var map = new google.maps.Map(document.getElementById("map-canvas"),
            mapOptions);

        // --- kode dibawah untuk menampilkan hasil query postgis berbentuk GeoJSON
---

        var xmlhttp = new XMLHttpRequest();
        var jsonstring = "";
        xmlhttp.onreadystatechange = function() {
          if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
            jsonstring = eval(xmlhttp.responseText);
            for (var h = 0; h < jsonstring.length; h++) {
              var lineStringCoords = [];
              var GeoJSON = jsonstring[h];
              for (var i = 0; i < GeoJSON.coordinates.length; i++) {
                lineStringCoords.push(new
google.maps.LatLng(GeoJSON.coordinates[i][1], GeoJSON.coordinates[i][0]));
              };
              var lineString = new google.maps.Polyline({
                path: lineStringCoords,
                map: map,
                strokeColor: getRandomColor(),
                strokeOpacity: 1.0,
                strokeWeight: 2
              });
            }
          }
        }
        xmlhttp.open("GET", "queryallroutes.php", true);
        xmlhttp.send();
      }

      google.maps.event.addDomListener(window, 'load', initialize);
    </script>
  </head>
  <body>
    <div id="map-canvas" />
  </body>
</html>
```