Visualisasi Data Histori KIRI pada Google Maps

Jonathan Laksamana Purnomo–2016730081

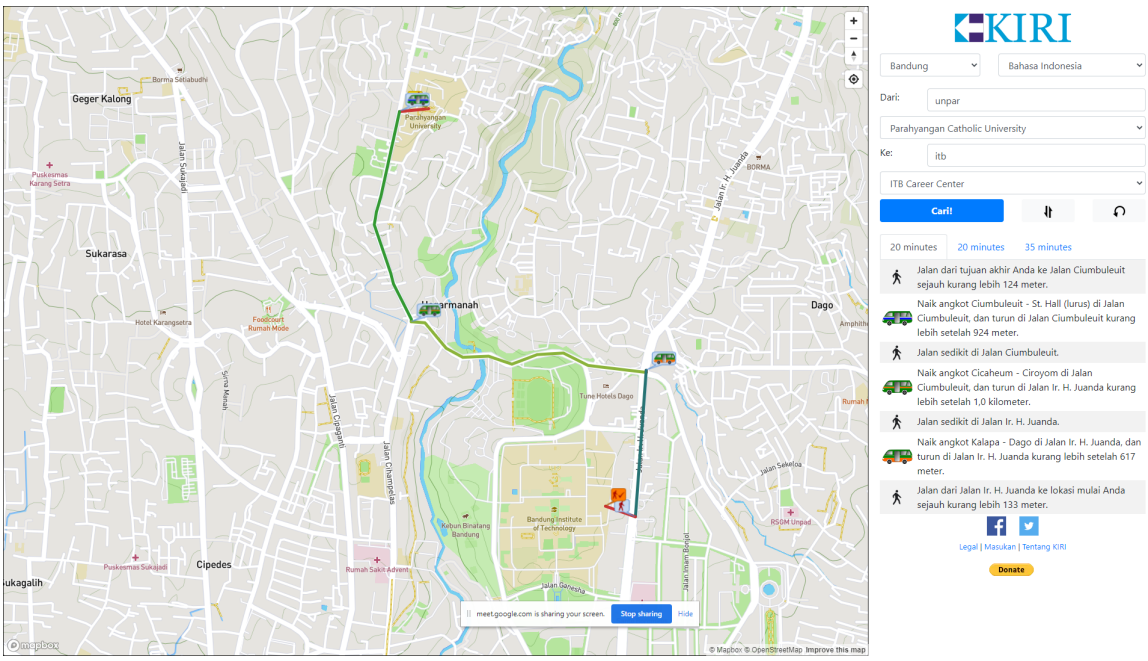
1 Data Skripsi

Pembimbing utama/tunggal: **Pascal Alfadian Nugroho**
Kode Topik : **PAN4992**
Topik ini sudah dikerjakan selama : **1** semester
Pengambilan pertama kali topik ini pada : Semester **45 - Ganjil 20/21**
Pengambilan pertama kali topik ini di kuliah : **Skripsi 1**
Tipe Laporan : **B** - Dokumen untuk reviewer pada presentasi dan **review Skripsi 1**

2 Latar Belakang

Kemajuan teknologi memudahkan manusia untuk mencari berbagai macam informasi. Salah satu informasi yang dapat diperoleh adalah informasi tentang navigasi transportasi publik. KIRI adalah perangkat lunak yang berguna sebagai navigasi antar kota menggunakan transportasi publik dengan menggunakan perangkat peta digital[?]. Pada awal pembuatannya KIRI dibuat untuk tujuan komersial. Namun karena dinilai kurang sukses, projek KIRI sekarang menjadi open source projek yang dapat di akses. Aplikasi KIRI memiliki beberapa fitur sebagai berikut:

- Pemilihan rute tercepat menggunakan angkutan kota.
- Menampilkan tempat pergantian angkutan kota.
- Memiliki fitur multi bahasa.
- Memiliki fitur pemilihan lokasi.
- Dapat menampilkan beberapa pilihan rute.
- Dapat menampilkan instruksi lengkap mencapai tujuan.



Gambar 1: Tampilan utama website KIRI

Pada perangkat lunak KIRI seluruh aktivitas yang dilakukan oleh user sudah tercatat. Data yang tercatat disebut juga dengan data histori. Data histori kiri memiliki jumlah record yang cukup banyak sehingga memungkinkan untuk mendapatkan informasi dari data tersebut. Tetapi data histori tersebut belum diolah secara maksimal.

Visualisasi Data adalah teknik untuk mengkomunikasikan data atau informasi dengan menggunakan objek visual seperti *graphic, chart, diagram, dll.* Salah satu objek visual yang dapat digunakan untuk merepresentasikan data adalah *Google Maps.*

Metode yang akan digunakan dalam memvisualisasikan data adalah *Heat Map* dan *Marker Clustering. Heat Map* adalah teknik visualisasi data yang menunjukkan besarnya suatu fenomena sebagai warna dalam dua dimensi. Sedangkan *Marker Clustering* adalah teknik visualisasi data yang mengelompokan *marker* atau *pointer* yang jarak *latitude* dan *longitude* nya saling berdekatan antara suatu *marker* dengan marker yang lainnya.

Pada skripsi ini akan dibangun perangkat lunak yang dapat memvisualisasikan data histori KIRI. Perangkat lunak ini akan menggunakan metode visualisasi *heat map* dan *marker clustering* dari hasil visualisasi tersebut akan diambil suatu pola kesimpulan dari data histori KIRI.

3 Rumusan Masalah

Rumusan masalah dari topik ini adalah sebagai berikut:

- Bagaimana memvisualisasikan data histori KIRI?
- Bagaimana menemukan pola dari data histori KIRI?
- Bagaimana penerapan metode *heat map* pada visualisasi data histori KIRI?
- Bagaimana penerapan metode *marker clustering* pada visualisasi data histori KIRI?

4 Tujuan

Tujuan dari topik ini adalah sebagai berikut:

- Mempelajari *Google Maps Javascript API.*
- Melakukan observasi data.
- Mengimplementasikan metode *Heat map* pada visualisasi data histori KIRI.
- Mengimplementasikan metode *Marker clustering* pada visualisasi data histori KIRI.

5 Detail Perkembangan Pengerjaan Skripsi

Detail bagian pekerjaan skripsi sesuai dengan rencan kerja/laporan perkembangan terkahir :

1. **Mempelajari atribut-atribut pada histori data KIRI.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Pada perangkat lunak KIRI seluruh aktivitas yang dilakukan oleh user sudah tercatat. Data yang tercatat disebut juga dengan data histori. Data histori kiri memiliki jumlah record yang cukup banyak sehingga memungkinkan untuk mendapatkan informasi dari data tersebut. Tetapi data histori tersebut belum diolah secara maksimal. Data histori KIRI memiliki format *comma separated values* (csv) yang memiliki tiga buah atribut yakni:

- timestamp adalah atribut yang mencatat waktu dan tanggal.
- start adalah atribut yang mencatat posisi awal pada data.
- end adalah atribut yang mencatat posisi tujuan pada data.

Perangkat lunak nantinya akan dapat mengkonversi data histori KIRI yang memiliki format csv menjadi format JSON agar dapat dilakukan proses pengolahan data.

2. **Mempelajari visualisasi data.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Visualisasi Data adalah teknik untuk mengkomunikasikan data atau informasi dengan menggunakan objek visual seperti *graphic, chart, diagram, dll.* Salah satu objek visual yang dapat digunakan untuk merepresentasikan data adalah *Google Maps.* Metode yang akan digunakan dalam memvisualisasikan data adalah *Heat Map* dan *Marker Clustering. Heat Map* adalah teknik visualisasi data yang menunjukkan besarnya suatu fenomena sebagai warna dalam dua dimensi. Sedangkan *Marker Clustering* adalah teknik visualisasi data yang mengelompokan *marker* atau *pointer* yang jarak *latitude* dan *longitude* nya saling berdekatan antara suatu *marker* dengan marker yang lainnya.

3. **Mempelajari *Google Maps API* khususnya *HeatMap* dan *MarkerClustering***
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Google Maps adalah layanan pemetaan web yang dikembangkan oleh Google. Menawarkan citra satelit, foto udara, dan peta jalan yang interaktif, kondisi lalu lintas secara *real time.* Dalam pengembangannya *google maps* memiliki akses pendukung untuk bahasa pemrograman *javascript.* Berikut ini beberapa layanan yang telah disediakan oleh *google maps javascript api[?]*:

   - *Maps.*
   - *Drawing Object.*
   - *Street Views.*
   - *Routes*

5.1 Map

Map adalah sebuah objek pada *google maps javascript api* yang digunakan untuk membuat objek *map* didalam *html* elemen. Untuk dapat menginisialisasi objek *map* pengembang harus dapat menggunakan constructor yang memiliki perintah: Map(mapDiv[, opts]) Parameters: mapDiv: Element opts: MapOptions optional Pada *constructor* diatas terdapat dua paramete:

- mapDiv.
- MapOptions

MapDiv adalah elemen html dimana objek map akan diinisialisasi, parameter ini bersifat wajib. MapOptions adalah sebuah objek yang dapat digunakan untuk mengatur *property* yang dimiliki oleh objek *map.* Berikut ini adalah contoh menginisialisasi objek *map*: function initMap() let mapOptions = center: lat: -6.914744, lng: 107.609810, zoom: 12, let map = new google.maps.Map(document.getElementById("map"), mapOptions); return map; Ketika fungsi *initMap()* dijalankan maka *google maps javascript api* akan membuat objek map didalam *html dom* yang memiliki id='map' dan akan mengatur *property* yang dimiliki sesuai dengan objek *mapOptions.*

Objek *map* telah menyediakan fungsi-fungsi yang bisa langsung digunakan oleh pengembang, beberapa fungsi yang disediakan oleh objek *map* adalah:

- Fungsi *fitBounds* berguna untuk menentukan *view port* dari objek *map* sesuai dengan *bounds* yang diberikan. Fungsi ini memiliki struktur perintah seperti: fitBounds(bounds[, padding]) Parameters: bounds: LatLngBounds|LatLngBoundsLiteral padding: number|Padding optional Return Value: None
- Fungsi *getBounds* berguna untuk mendapatkan *view port* dari objek *map*. Fungsi ini memiliki struktur perintah seperti: getBounds() Parameters: None Return Value: LatLngBounds
- Fungsi *getCenter* berguna untuk mendapatkan posisi yang ditunjukan pada objek *map* posisi yang didapatkan akan berbentuk *longtitude dan langtitude*. Fungsi ini memiliki struktur perintah seperti: getCenter() Parameters: None Return Value: LatLng
- Fungsi *getClickableIcons* berguna untuk mendapatkan *clickable icon* setiap *clickable icon* merupakan *point of interest* pada *map* . Fungsi ini memiliki struktur perintah seperti: getClickableIcons() Parameters: None Return Value: boolean
- Fungsi *getMapTypeId* beguna untuk mendapatkan *id* dari jenis *map* yang digunakan . Fungsi ini memiliki struktur perintah seperti: getMapTypeId() Parameters: None Return Value: MapTypeId|string *Google Maps Javascript API* menyediakan variabel konsanta / *constant* yang berfungsi untuk menentukan jenis peta yang akan digunakan pada objek *map*. Konsanta ini dapat memiliki empat nilai yaitu:
  - HYBRID jenis peta ini akan menampilkan layar *transparan* pada jalan-jalan utama pada citra satelit.
  - ROADMAP jenis peta ini akan menampilkan *street map*.
  - SATELLITE jenis peta ini akan menampilkan citra satellite.
  - TERRAIN jenis peta ini akan menampilkan bentuk nyata dari kondisi geologi suatu tempat.
- Fungsi *setMapTypeId* beguna untuk membuat atau mengubah *mapTypeId*. Fungsi ini memiliki struktur perintah seperti: setMapTypeId(mapTypeId) Parameters: mapTypeId: MapTypeId|string
- Fungsi *setZoom* berguna untuk mengubah *zoom value* yang dimiliki oleh objek *map*. setZoom(zoom) Parameters: zoom: number
- Fungsi *setMapOption* berguna untuk mengubah *property mapOption* yang dimiliki oleh objek *map*. setOptions(options) Parameters: options: MapOptions

5.2 Sistem Kordinat Google Maps

Google Maps Javascript API menggunakan kelas *LatLng* untuk merepresentasikan kordinat secara geografis pada objek *map*. Kelas *LatLng* merupakan sebuah kelas yang merepresentasikan latitude dan longtitude

- Latitude memiliki batas antara -90 sampai 90 derajat, jika ada nilai yang diluar batasan tersebut maka nilai tersebut akan dibulatkan kebatas terdekat.
- Longtitude memiliki batas antara -180 sampai 180 derajat, jika ada nilai yang diluar batasan tersebut maka nilai tersebut akan dibulatkan kebatas terdekat.

Untuk dapat menginisialisasi kelas *LatLng* pada *google maps javascript api* pengembang perlu membuat *constructor* yang memiliki struktur: LatLng(lat, lng[, noWrap]) Parameters: lat: number lng: number noWrap: boolean optional Membuat objek LatLng yang mewakili titik geografis. *latitude* ditentukan dalam derajat dalam rentang [-90, 90]. *longtitude* ditentukan dalam derajat dalam rentang [-180, 180]. Set *noWrap true* untuk mengaktifkan nilai di luar rentang ini. Contoh penggunaan kelas *LatLng*:

map.setCenter(new google.maps.LatLng(-34, 151)); map.setCenter(lat: -34, lng: 151); *Google Maps Javascript API* telah menyediakan fungsi bawaan yang dapat diakses ketika menggunakan kelas *LatLng*. Fungsi tersebut adalah:

- Fungsi *equals* fungsi ini bertujuan untuk membandingkan posisi antara objek *map*. equals(other) Parameters: other: LatLng Return Value: boolean
- Fungsi *lat* fungsi ini bertujuan untuk mendapatkan posisi *latitude* dari objek *map*. lat() Parameters: None Return Value: number Returns the latitude in degrees.
- Fungsi *lng* fungsi ini bertujuan untuk mendapatkan posisi *longtitude* dari objek *map*. lng() Parameters: None Return Value: number Returns the longitude in degrees.
- Fungsi *toJSON* fungsi ini akan mengembalikan format *json* terhadap posisi latlng padda objek *map*
- Fungsi *toUrlValue* Mengembalikan string dalam bentuk "lat, lng" untuk LatLng ini.

5.3 Ovelay System

Pada subbab ini akan dijelaskan cara penggunaan kelas *overlay* yang disediakan oleh *google maps javascript api*. *Google Maps Javascript API* memiliki kelas-kelas yang digunakan untuk menggambar suatu objek didalam objek *map*. Beberapa kelas *overlay* yang disediakan adalah sebagai berikut:

- *Marker*.
- *Heat Map*.

5.4 Marker

Gambar 2: Add Marker

Marker adalah sebuah kelas yang dapat memunculkan *mark* / tanda pada objek *map*. Untuk dapat menginisilasisasi kelas *marker* pengembang dapat menggunakan *constructor* yang memiliki struktur seperti: Marker([opts]) Parameters: opts: MarkerOptions optional Pada *constructor marker* terdapat parameter *markeroptions* yang bersifat optional. *MarkerOptions* merupakan sebuah objek yang dapat digunakan pada objek *mark*. *MarkerOptions* memiliki atribut seperti:

- Atribute *anchorPoint*.
- Atribute *animation*.
- Atribute *clickable*.

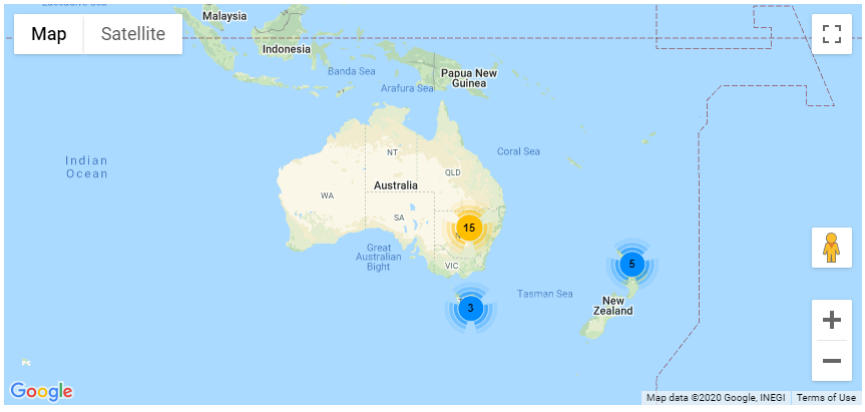
- Atribute *crossOnDrag*.
- Atribute *cross*.
- Atribute *dragable*.
- Atribute *icon*.
- Atribute *label*.
- Atribute *map*.
- Atribute *opacity*.
- Atribute *position*.

Contoh penginisialisasian *marker* pada objek *maps* adalah: return new google.maps.Marker(position: location, label: labels[i); Kelas *marker* memiliki fungsi bawaan yang telah disediakan oleh antara lain seperti:

- getAnimation.
- getClickable.
- getCursor.
- getDraggable.
- getIcon.
- getMap.
- getOpacity.
- getPosition.
- getShape.
- getTitle.

5.5 Marker Clusterer

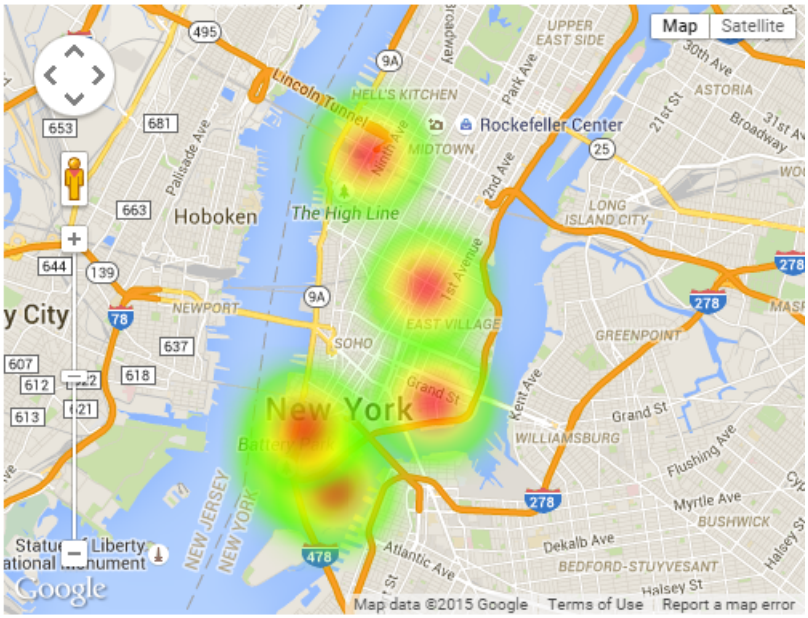
Google Maps Javascript API telah menyediakan kelas *MarkerClustererPlus* untuk dapat mengabungkan objek *marker* 5.4.



Gambar 3: Contoh Marker Clustering

Untuk dapat menginisialisasi kelas tersebut pihak pengembang perlu menuliskan perintah: var markerCluster = new MarkerClusterer(map, markers, imagePath: '*path/m*'); *Marker Clusterer* sendiri meruapakan suatu library untuk kelas *mark* sehingga kelas ini dapat menggunakan fungsi turunan dari kelas *mark*.

5.6 HeatMap



Gambar 4: Contoh HeatMap

Google Maps Javascript API telah menyediakan kelas *heatmap* untuk menampilkan *heatmap* pada objek *maps*. Untuk dapat menginisialisasi kelas ini pengembang perlu memanggil perintah *constructor* yang memiliki struktur seperti: HeatmapLayer([opts]) Parameters: opts: HeatmapLayerOptions optional .

Kelas *HeatMap* dilengkapi dengan parameter *HeatmapLayerOptions* yang bersifat opsional, objek ini bertujuan untuk dapat mengatur *property* dari kelas *HeatMap*. Parameter *HeatMapLayerOptions* merupakan sebuah objek yang memiliki atribut sebagai berikut:

- data *titik-titik* data yang diperlukan.
- dissipating variabel yang menentukan apakah *heatmap* akan menghilang jika *maps* diperbesar atau diperkecil.
- gardient gardient dari warna *heatmap*
- map atribute untuk menunjukan peta dimana *heatmap* akan ditampilkan.
- maxIntencity nilai maximal dari intensitas warna pada *heatmap*.
- opcity nilai opacity dari *heatmap*.
- radius nilai radius dari *heatmap*.

Contoh penginisialisasian kelas *heatmap*: let heatmap = new google.maps.visualization.HeatmapLayer(data: heatmapData); Kelas *HeatMap* memiliki fungsi-fungsi bawaan yang telah disediakan oleh *google maps* beberapa fungsi tersebut adalah:

- getData() fungsi ini akan mengembalikan data point pada objek *heatmap*.
- getMap() fungsi ini akan mengembalikan objek *map*.
- setData(data) fungsi ini akan memasukan data pada objek *heatmap*.
- setMap(map) fungsi ini akan memasukan objek *map* pada objek *heatmap*.
- setOption(option) fungsi ini akan memasukan objek *HeatMapLayerOptions* pada objek *heatmap*.

4. **Membuat desain perangkat lunak**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Belum dikerjakan.

5. **Mengimplementasi perangkat lunak**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Belum dikerjakan.

6. **Menulis dokumen skripsi**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Belum dikerjakan.

6 Pencapaian Rencana Kerja

Langkah-langkah kerja yang berhasil diselesaikan dalam Skripsi 1 ini adalah sebagai berikut:

1. Mempelajari atribut-atribut pada histori data KIRI.
2. Mempelajari visualisasi data.
3. Mempelajari *Google Maps API* khususnya *HeatMap* dan *MarkerClustering*.

Bandung, 01/15/2021

Jonathan Laksamana Purnomo

Menyetujui,

Nama: Pascal Alfadian Nugroho
Pembimbing Utama