Intelligent Maritime Transportation System: Visualisasi Data Kapal Berbasis AIS Menggunakan Peta Daring

Oleh: Shidqon Famulaqih

Pembimbing 1

Dr. I Ketut Eddy Purnama, S.T., M.T.

Pembimbing 2

Christyowidiasmoro, S.T., M.T.

Outline

PENDAHULUAN

Latar Belakang
Rumusan Masalah
Tujuan
Manfaat
Batasan Masalah

TINJAUAN PUSTAKA

PEMBAHASAN

Desain Sistem dan Implementasi
Pengujian

PENUTUP

Kesimpulan
Saran

Latar Belakang

1. Data IMTS dibutuhkan untuk memonitoring kapal
2. Data IMTS adalah data dari AIS dengan jumlah yang begitu banyak dan memiliki jenis yang beragam
3. Informasi susah ditangkap dari data mentah

Perumusan Masalah

1. Data mentah dari IMTS tidak memberi informasi yang cukup
2. Diperlukan visualisasi untuk menampilkan data untuk menyampaikan informasi secara jelas

Tujuan & Manfaat

Tujuan :
Melakukan visualisasi data IMTS menggunakan peta daring.

Manfaat:
Hubungan antar data IMTS bisa ditampilkan secara jelas untuk mempermudah penangkapan informasi

Batasan Masalah

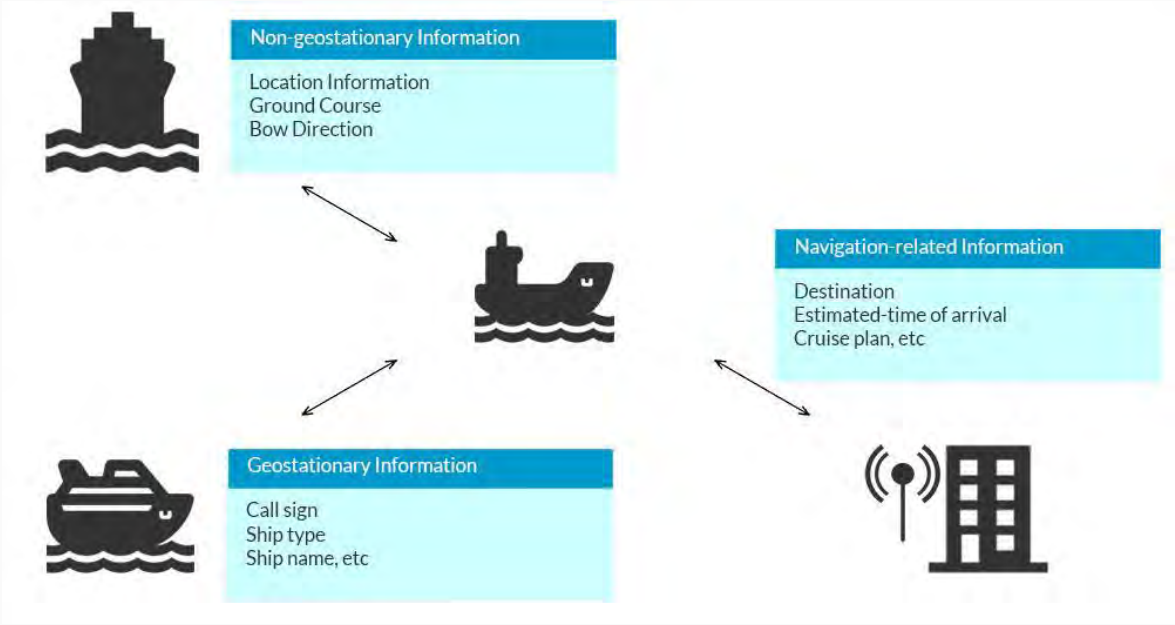
1. Data yang divisualisasikan adalah data IMTS dari AIS yang sudah dalam bentuk database
2. Visualisasi data IMTS dalam bentuk peta daring menggunakan Google Maps V3 API

Tinjauan Pustaka

1. AIS Data
2. Google Maps API
3. Pemrograman MVC
4. REST API

Tinjauan Pustaka

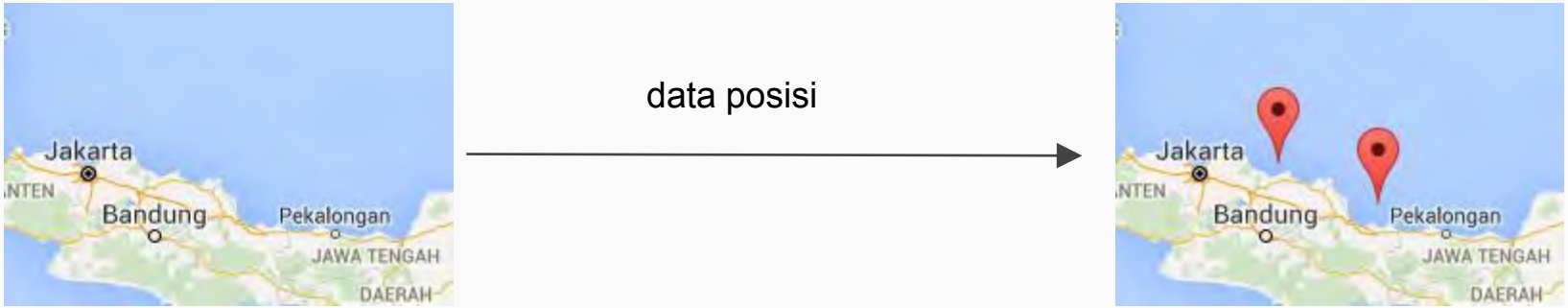
AIS DATA



Tinjauan Pustaka

Google Maps API

Memungkinkan pengembang menggunakan peta google untuk menampilkan data terkait posisi



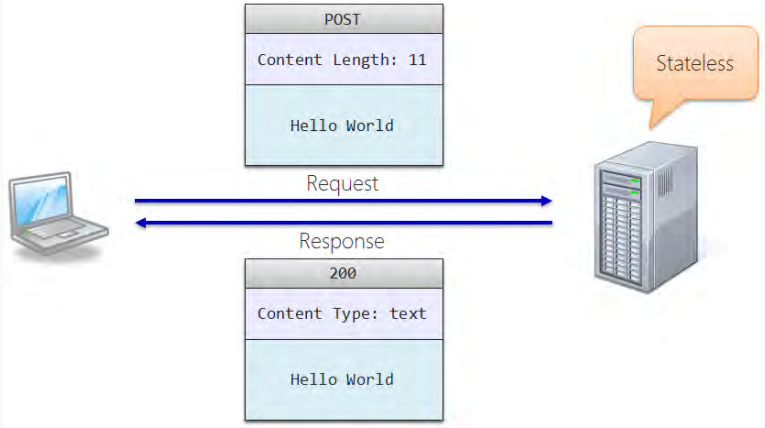
Tinjauan Pustaka

Pemprograman MVC adalah pemrogram terstruktur dengan memisah bagian dari program sesuai fungsinya yaitu Model, View dan Controller.

1. Model bertugas mengolah data
2. View bertugas menampilkan data
3. Contoller bertugas mengatur kapan data ditampilkan

REST (*Representational State Transfer*) API

Metodologi yang dirancang agar program dapat berkomunikasi satu sama lain dengan cara yang sesederhana mungkin melalui HTTP protokol

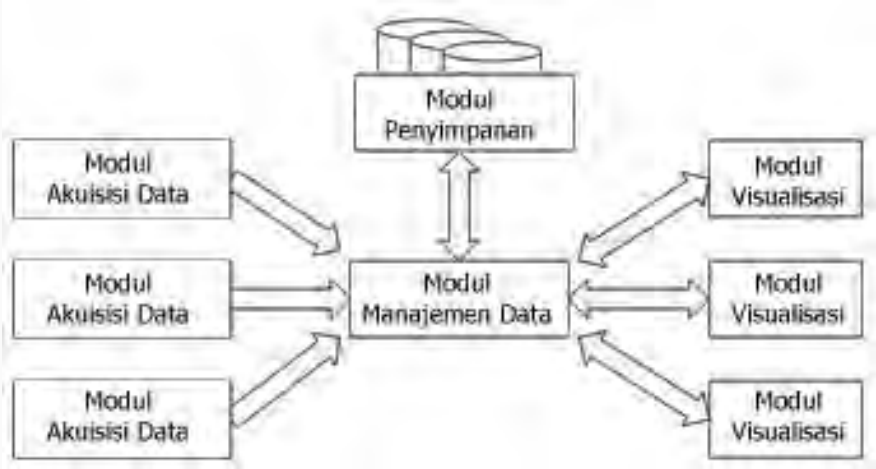


Transfer data dalam format XML atau JSON

Desain Sistem

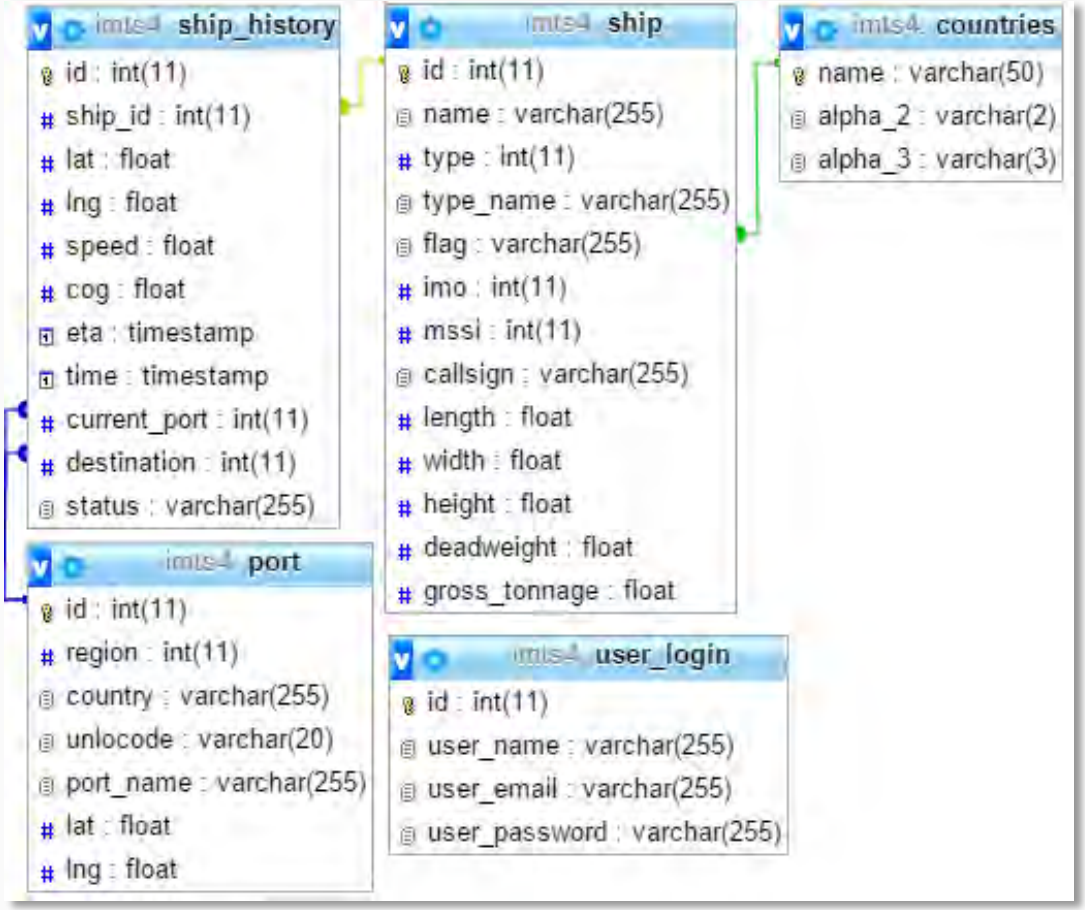
Desain IMTS terdiri dari 4 modul:

1. Akuisisi Data
2. Manajemen Data
3. Penyimpanan Data
4. Visualisasi Data



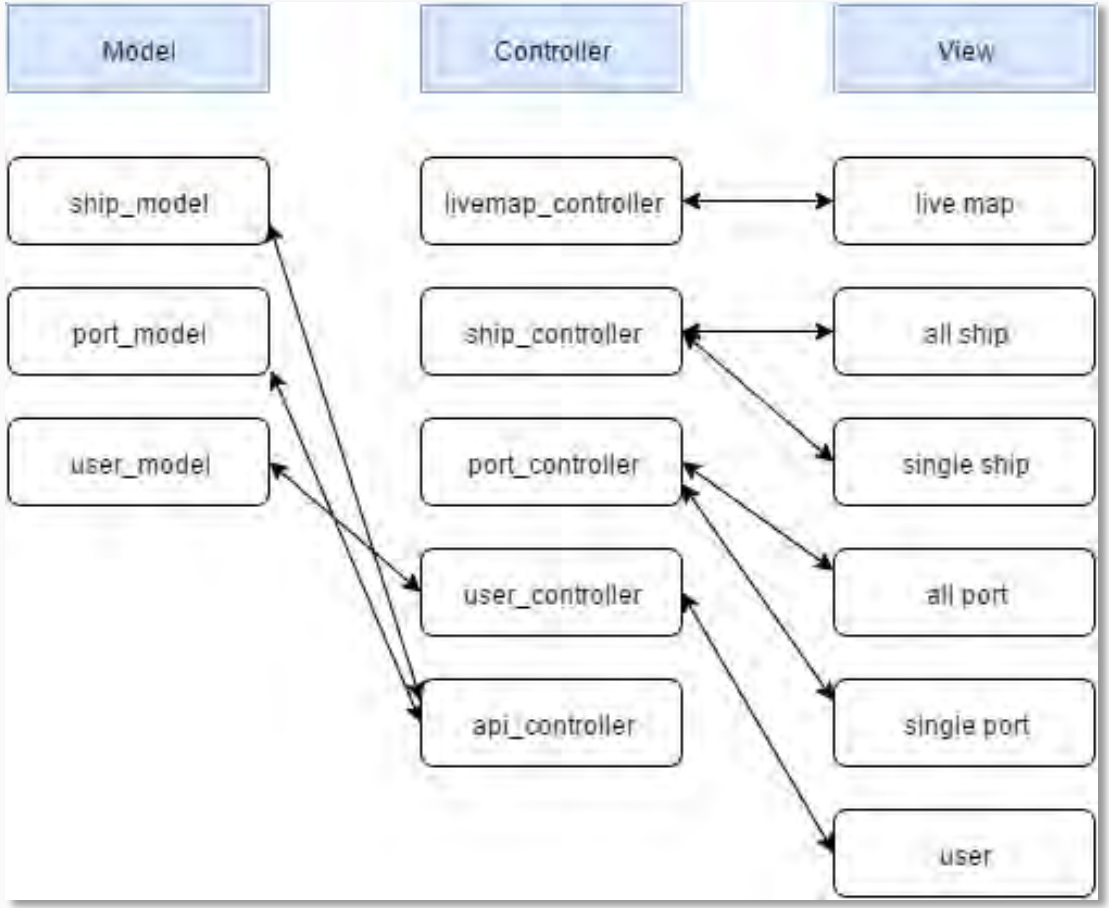
Modul Penyimpanan Data

Data IMTS disimpan pada database berbasis MySQL

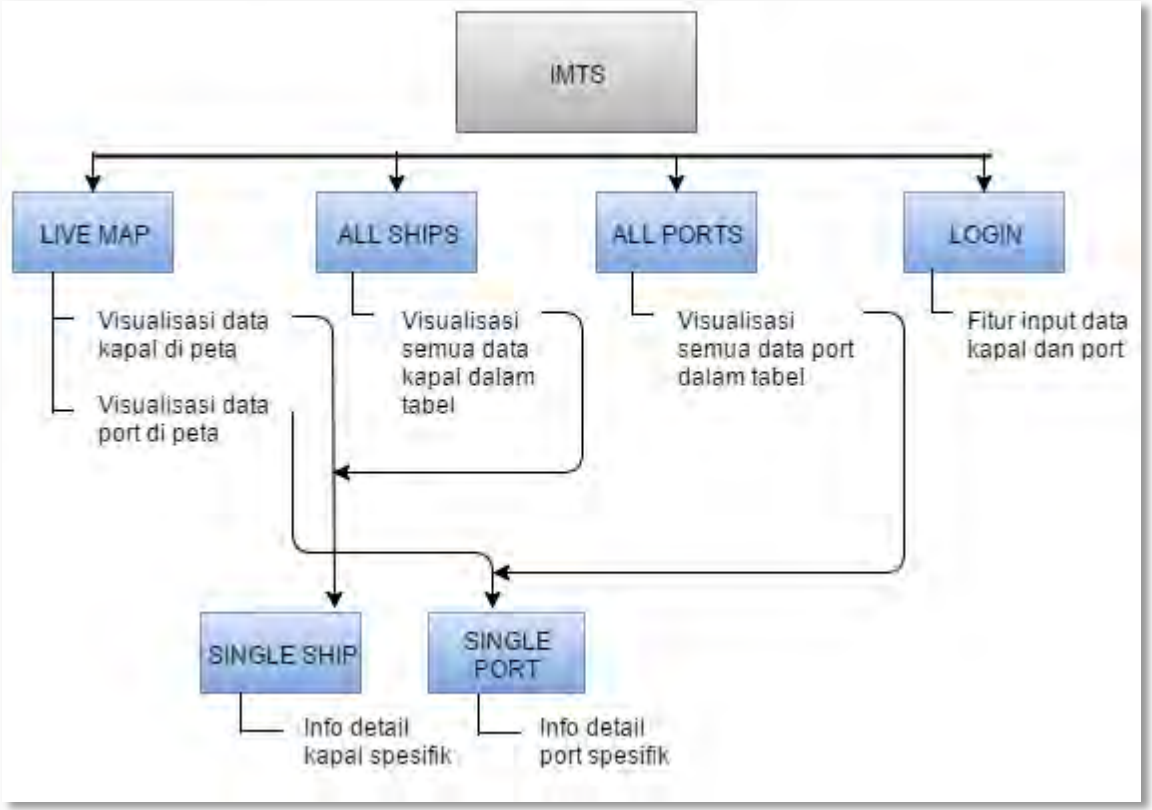


Modul Manajemen Data dan Visualisasi Data

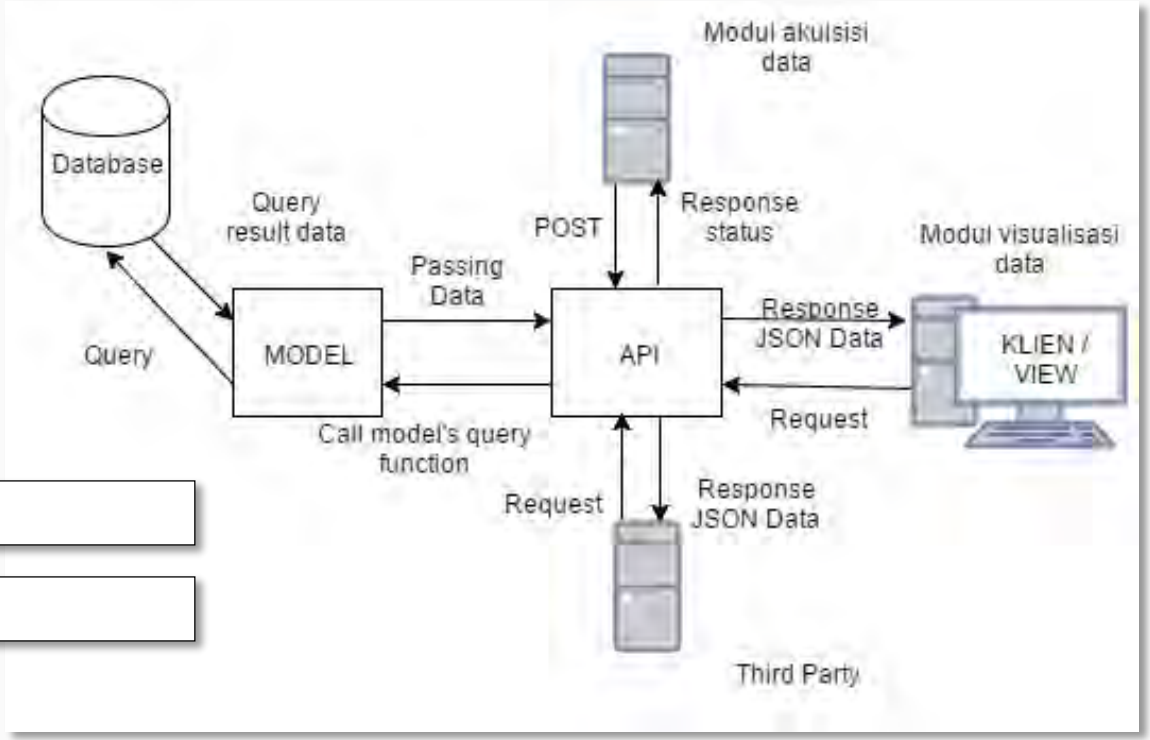
Dibuat dengan stuktur MVC



Bagian View


IMTS
LIVE MAP
ALL SHIPS
ALL PORTS
LOGIN
Visualisasi data kapal di peta
Visualisasi data port di peta
Visualisasi semua data kapal dalam tabel
Visualisasi semua data port dalam tabel
Fitur input data kapal dan port
SINGLE SHIP
SINGLE PORT
Info detail kapal spesifik
Info detail port spesifik

Desain REST API




```
http://server_address/api/ships
```

```
http://server_address/api/ports
```

Parameter POST
Data Spesifikasi Kapal

Parameter	Nilai	Description
data	1	POST untuk data spesifikasi kapal
name	string	Nama kapal
imo	integer	IMO
mssi	integer	MSSI
callsign	string	Callsign
flag	string	Kode negara
type	string	Tipe kapal
deadweight	float	Deadweight
gross_tonnage	float	Gross Tonnage

```
POST http://server_address/api/ships?data=1&name=name&imo=imo&mssi=mssi&...
```

Parameter POST
Data Riwayat Kapal

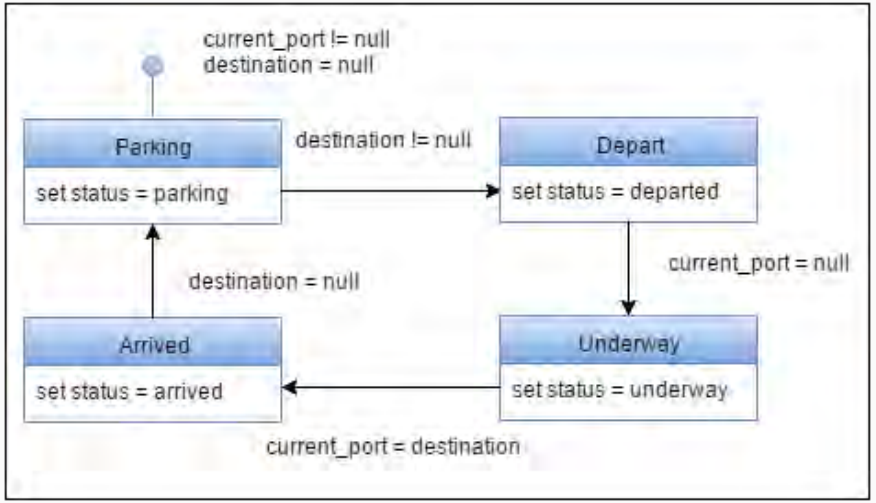
Parameter	Nilai	Description
history	1	POST untuk data riwayat kapal
id	integer	ID unik kapal
lat	float	Latitude kapal
lng	float	Longtitude kapal
speed	float	Kecepatan kapal
current_port	integer	ID unik port tempat kapal berada
destination	integer	ID unik port tujuan kapal

```
POST http://server_address/api/ships?history=1&id=id&lat=lat&lng=lng&...
```

Data tambahan

Beberapa tambahan diolah terlebih dahulu sebelum disimpan ke database

1. Course
2. Current Port
3. Status Pergerakan kapal



Parameter GET
Data Kapal / PORT

Parameter	Nilai	Description
id	integer	ID unik kapal / port
lat1	float	Posisi viewport peta
lat2	float	
lng1	float	
lng2	float	
portcall (khusus port)	“ship_in”, “arrivals”, ”departures”, “expected_arrivals”	Status kapal terhadap port

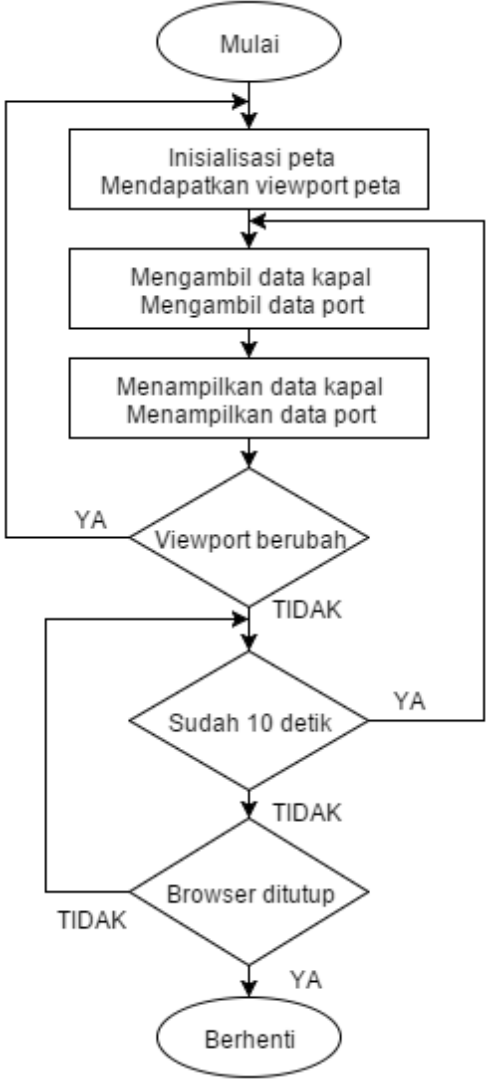
Contoh respon:

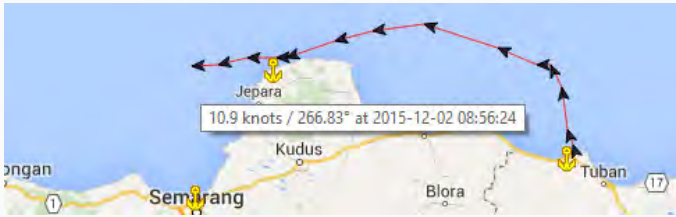
```
[{
  "id" : integer,
  "name" : string,
  "lat" : float,
  "lng" : float,
  ...
   // data selengkapnya
 },
 {
  ...
  // data kapal / port selanjutnya
}]
```

```
POST http://server_address/api/ships?history=1&id=id&lat=lat&lng=lng&...
```

DESAIN VISUALISASI

1. Marker Kapal
2. Marker Port
3. Marker Cluster
4. Marker Track





Pengujian

1. Pengujian API
2. Pengujian Visualisasi

Pengujian API

Untuk mengetahui apakah memberikan response API sesuai dengan yang didesain untuk setiap request yang dikirim

No	Parameter	Response
1	lat1 = -5.6350 lng1 = 114.7518 lat2 = -8.3609 lng2 = 107.2481	Menghasilkan respon 6 data kapal.
2	lat1 = 2.0554 lng1 = 116.3256 lat2 = 0.4353 lng2 = 112.573	Menghasilkan respon not found untuk peta yang menampilkan daerah daratan di tengah pulau Kalimantan
3	lat1 = null lng1 = 104.7379 lat2 = 0.8567 lng2 = 102.8620	Menghasilkan respon status error karena ada parameter viewport yang kosong (lat1).
4	lat1 = 5.0825 lng1 = 121.5848 lat2 = abcd lng2 = 114.0811	Menghasilkan respon status error karena ada parameter viewport yang diisi string (lat2).

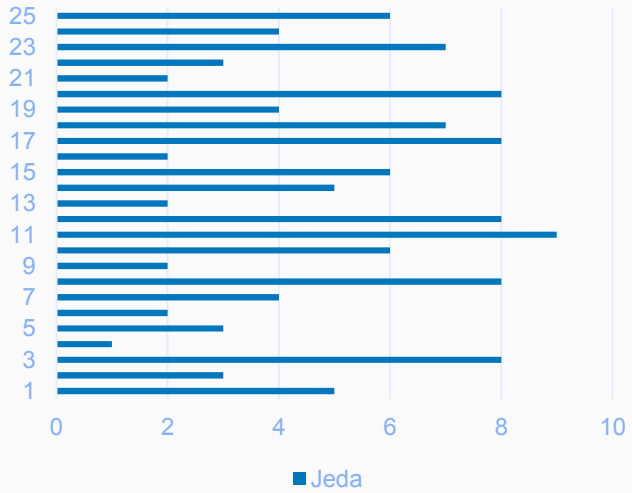
Pengujian API

..continue

No	Parameter	Response
5	lat1 = 4.3654 lng1 = 190.6540 lat2 = -2.1117 lng2 = 115.6467	Menghasilkan respon status error karena ada parameter viewport yang melebihi jangkauan nilai latitude dan longtitude (lng1)..
6	id = 3380	Menghasilkan respon 1 data kapal
7	id = KRI	Menghasilkan respon status error karena parameter id diisi string
8	id = null	Menghasilkan status error karena parameter id kosong

Pengujian Visualisasi

1. Untuk mengetahui jeda rata-rata penampilan data terbaru pada peta daring.
2. Dilakukan dengan melakukan POST data riwayat posisi kapal melalui API. Jeda penampilan dihitung dari kapan data terbaru dikirim sampai ditampilkan



Kesimpulan

1. Penggunaan peta daring membuat data kapal yang banyak bisa ditampilkan secara efisien dalam 1 tampilan secara bersamaan.
2. Terdapat jeda waktu dengan rata-rata 4,92 detik dalam menampilkan data terbaru ketika ada data baru yang diterima.

Saran

1. Data yang diuji tergolong masih sedikit. Untuk jumlah yang besar diperlukan *database scalability* untuk mempercepat pengolahan data.

2. Untuk mengurangi jeda waktu dalam menampilkan data terbaru, aplikasi bisa diatur untuk melakukan muat ulang data terbaru secara otomatis menjadi lebih cepat dari 10 detik. Namun hal ini akan memberatkan pengolahan disisi *client* sehingga diperlukan optimasi dibagian proses penampilan *view*.

TERIMA KASIH