Problem Statement

A vehicle fleet company has telematic devices fitted in their vehicles. These devices emit constant data which need to be sent to the backend. The data emitted is as below

- Latitude and longitude of the vehicle at the given time
- Percent of fuel left at the given time
- Vehicle speed at the given time

You need to design the backend system which

- Can receive and save the data from the vehicles telematics devices
- Generate these reports from this data

  Total distances covered by each vehicle - daily /monthly

  How many vehicles did overspeeding & details of each overspeeding vehicle - daily/monthly

- Design the system in such a way that 3rd party developers can leverage the APIs of this platform for building new capabilities around the data available.

- Design the system to have monitoring and execute a set of actions based on rules (for e.g. send a notification if speed is above 80 kmph). System should be able to configure different actions and rules.

Introduction

This is description of a backend platform that is capable of receiving data from vehicles registered with the services. It exposes a bunch of APIs for vehicle registration and publishing of data. It uses Kafka for message exchange. The overall design looks as below
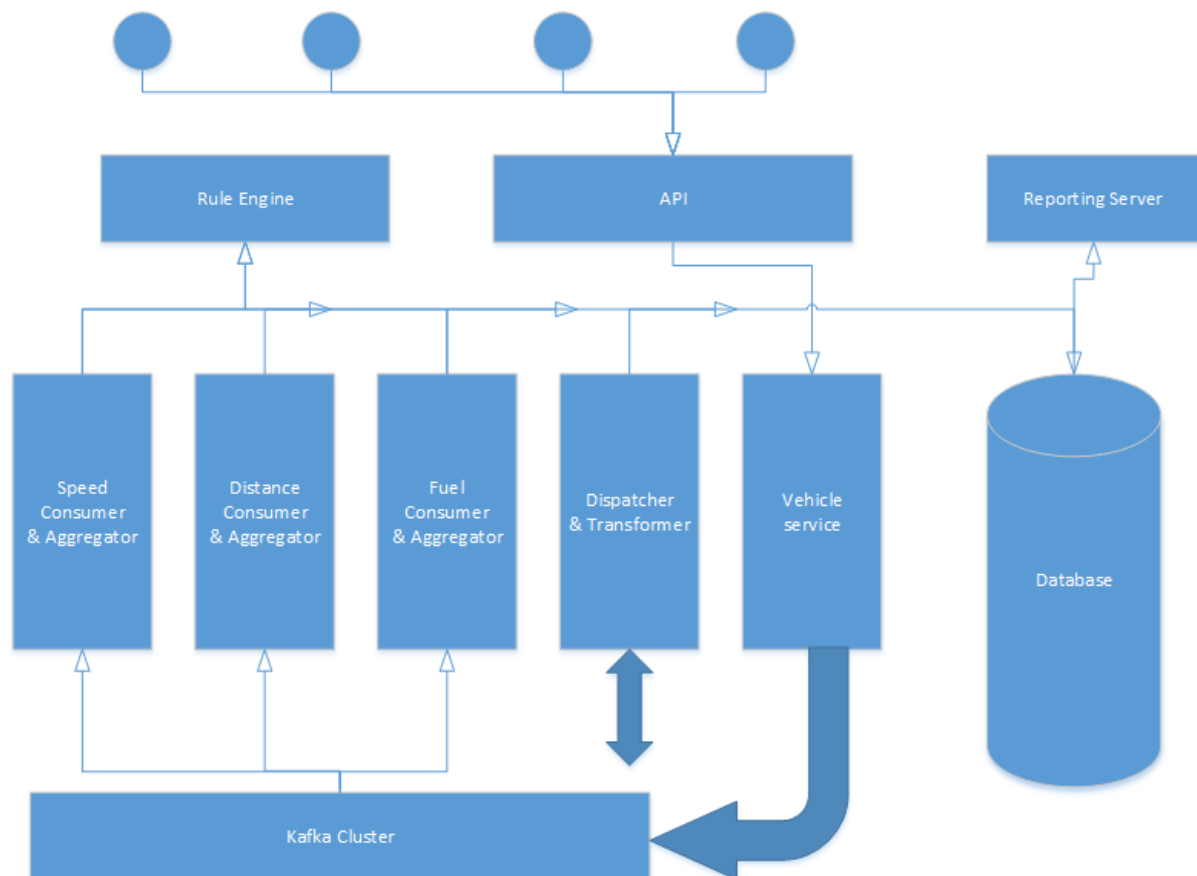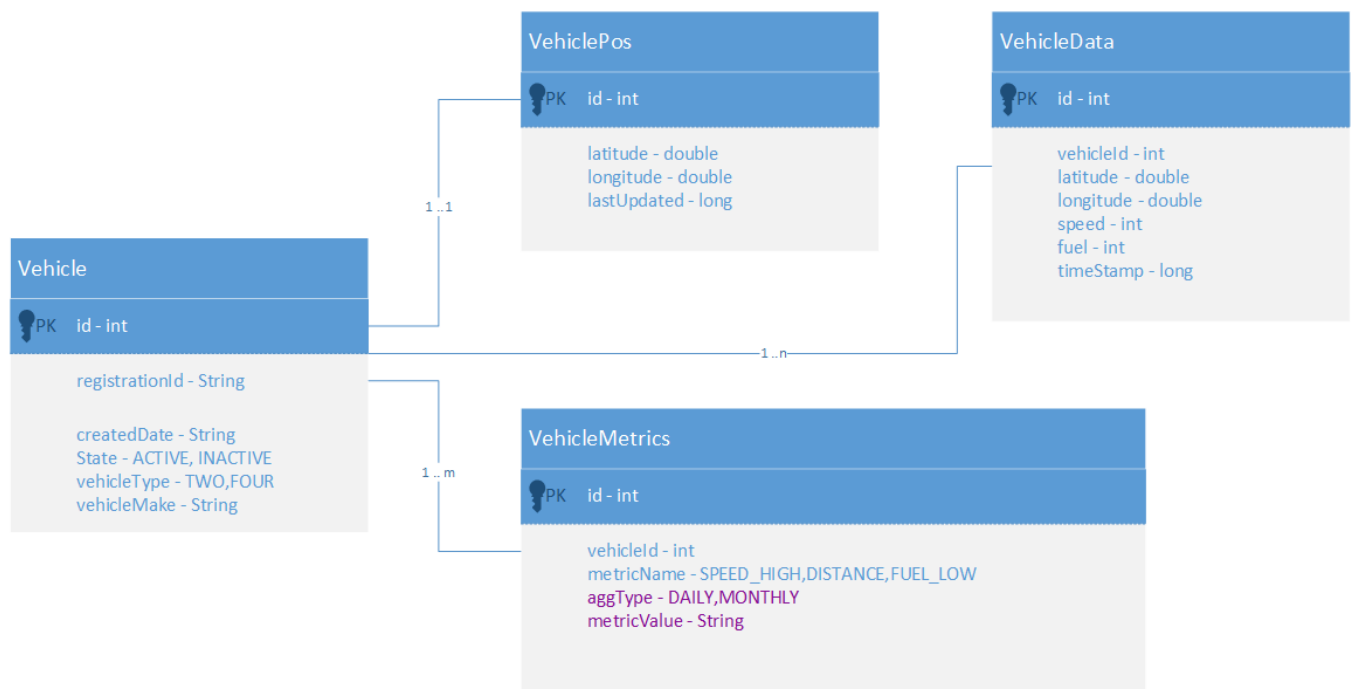

Assumptions

1. Vehicles are capable of sending their location, speed and fuel information at an periodic interval of 5 mins, either themselves or via a proxy.
2. Vehicles will be registering themselves with the backend service and will be allocated an unique id. They should be capable of sending this id along with the data.
3. Haversine formula is used to calculate the distance travelled based on the latitude. It is assumed to be relatively accurate to calculate short distances.

APIs

Method	Url	Payload

Post	http://localhost:8080/api/vmf/v1/vehicle	
		``json
		{
		    "registrationId":"ckm8812",
		    "vehicleType":"TWO",
		    "makeType":"Scooter"
		}
		``

Post	http://localhost:8080/api/vmf/v1/vehicle/{id}/metric	
		``json
		{
		    "id": 1,
		    "latitude":30.741482,
		    "longitude":76.768066,
		    "fuel": 50,
		    "speed": 150,
		    "timeStamp": 1677334746000
		}
		``

Get	http://localhost:8080/api/vmf/v1/vehicle	NA
Get	http://localhost:8080/api/vmf/v1/vehicle/{id}/metric	NA

Data Model

VehiclePos

PK id - int

latitude - double
longitude - double
lastUpdated - long

VehicleData

PK id - int

vehicleId - int
latitude - double
longitude - double
speed - int
fuel - int
timeStamp - long

Vehicle

PK id - int

registrationId - String

createdDate - String
State - ACTIVE, INACTIVE
vehicleType - TWO,FOUR
vehicleMake - String

VehicleMetrics

PK id - int

vehicleId - int
metricName - SPEED_HIGH,DISTANCE,FUEL_LOW
aggType - DAILY,MONTHLY
metricValue - String

1..1

1..n

1..m

Technologies Used

Springboot, Java11, Kafka, H2 database, REST, Jasper reports, Jeasy rule engine

Scale Calculations

A typical request takes about 10ms until the metric is pushed into Kafka.

Optimum threadpool size in embedded Tomcat – 100

Total metrics that can be published to kafka in 1 second – 100 * 100 = 10000 rps

It is taking about 43 ms for data aggregation and rule verification

For each metric -> 14.33 ms

This gives an rps of 70 rps.

Few ways that this can be improved –

1. Use of cache, this can bring down the time taken for aggregation considerably.
2. Use of Kafka Streams, since Kafka is write optimized we can achieve higher rps.

Storage requirements –
Assuming a 5 mins interval in sending data, in 24 hrs a vehicle can send 288 times.
A single metric is 150 bytes. 288 * 150 – 43KB
In a month ~ 1.3Mb per vehicle. For 10k vehicles = 13GB

References

https://en.wikipedia.org/wiki/Haversine_formula

https://github.com/j-easy/easy-rules