Mục lục

[MỞ ĐẦU 3](#_Toc500540453)

[1.1 Một số khái niệm cơ bản GPS 4](#_Toc500540454)

[1.1.1 Lịch sử GPS 4](#_Toc500540455)

[1.1.2 Cấu trúc hệ thống GPS 5](#_Toc500540456)

[1.1.3 Cấu trúc tín hiệu GPS 6](#_Toc500540457)

[1.1.4 Các trị đo GPS 7](#_Toc500540458)

[1.2 Tình hình GPS trong nước và ngoài nước 7](#_Toc500540459)

[1.2.1 Tình hình trong nước 7](#_Toc500540460)

[1.2.2 Tình hình ngoài nước 8](#_Toc500540461)

[1.2.3 Tổng kết 8](#_Toc500540462)

[CHƯƠNG 2: HỆ THỐNG GPS VÀ ỨNG DỤNG TRACCAR. 9](#_Toc500540463)

[2.1 Xây dựng hệ thống GPS 9](#_Toc500540464)

[2.1.1 Thiết bị GPS GT02: 9](#_Toc500540465)

[2.1.1.1Thông số kỹ thuật 9](#_Toc500540466)

[2.1.1.2 Trạng thái thiết bị làm việc 9](#_Toc500540467)

[2.1.1.3 Cài đặt thẻ SIM: 10](#_Toc500540468)

[2.2 Giới thiệu hệ thống server TRACCAR: 10](#_Toc500540469)

[2.2.1Cài đặt hệ thống: 10](#_Toc500540470)

[2.2.2 Sơ đồ hoạt động của server: 11](#_Toc500540471)

[2.2.3 Thiết bị kết nối với server: 11](#_Toc500540472)

[2.2.4 Server thêm thiết bị: 12](#_Toc500540473)

[2.2.5 Cơ chế hoạt động của server: 13](#_Toc500540474)

[2.2.6 Hệ thống thông báo: 13](#_Toc500540475)

[2.2.7 Thiết lập Cơ sở dữ liệu: 14](#_Toc500540476)

[2.2.7.1 Thay đổi cơ sở dữ liệu: 14](#_Toc500540477)

[2.2.7.2 Cơ sở dữ liệu. 14](#_Toc500540478)

[2.2.8Biểu đồ phân rã chức năng. 20](#_Toc500540479)

[CHƯƠNG 3: CHƯƠNG TRÌNH QUẢN LÝ XE 21](#_Toc500540480)

[3.1 Giới thiệu chương trình quản lý xe. 21](#_Toc500540481)

[3.2 Chức năng chương trình. 23](#_Toc500540482)

[3.2.1 Phân quyền user: 23](#_Toc500540483)

[3.2.2 Thiết kế giao diện chương trình. 25](#_Toc500540484)

[3.2.2.1 Giới thiệu framework Ext JS 25](#_Toc500540485)

[3.2.2.2 Component trong Ext JS 25](#_Toc500540486)

[3.2.2.4 Kiến trúc MVC 27](#_Toc500540487)

[3.2.2.5 Phát triển giao diện 28](#_Toc500540488)

[3.2.3 Quản lý biển số xe. 29](#_Toc500540489)

[3.2.3.1 Tìm hiểu cách thức hoạt động của chương trình 29](#_Toc500540490)

[3.2.3.1 Thêm thuộc tính biển số xe vào database 30](#_Toc500540491)

[3.2.3.2 Truy vấn dữ liệu và hiển thị liên giao diện 30](#_Toc500540492)

[3.2.4 Thông báo ngày hết hạn thiết bị. 31](#_Toc500540493)

[3.2.5 Chế độ xem lại. 33](#_Toc500540494)

[3.2.5.1 Giới thiệu 33](#_Toc500540495)

[3.2.5.2 Tìm hiểu API và cách request api 34](#_Toc500540496)

[3.2.5.3 Requestr API TracCar 35](#_Toc500540497)

[KẾT LUẬN 40](#_Toc500540498)

[**TÀI LIỆU THAM KHẢO** 41](#_Toc500540499)

# MỞ ĐẦU

Trong xu thế CNTT hiện nay GPS đang là 1 hướng có tiềm năng rất lớn mang lại các dịch vụ hữu ích và giá trị cao. Nó là 1 nền tảng quan trọng trong các dịch vụ dựa vị trí (LBS).Việc xây dựng các ứng dụng trên nó đòi hỏi rất nhiều công sức và mang tính hệ thống cao. Có thể dẫn chứng cho xu thế này đó là việc đã và đang xuất hiện hàng loạt các hệ thống theo dõi, chỉ đường cho Taxi, Xe Tải,.., các bản đồ : đất đai, quy hoach, địa điểm trực tuyến. Và thậm chí là mạng xã hội dựa vị trí. Quan trọng nhất vẫn là các dịch vụ dựa trên nó. Ví dụ như : tìm kiếm địa điểm, tìm đường, tìm bạn bè, tra cứu thông tin...

Trong 1 hệ thống vận tải : Taxi, Xe tải, Xe Khách, Xe buýt ... thì việc theo dõi,quản lý điều hành là 1 điều tất yếu. Trước đây, Việc quản lý đa phần dựa vào việc ghi lại lịch trình của lái xe và người quản lý. Mang tính chủ quan và thiếu chuyên nghiệp, hơn nữa đòi hỏi lượng nhân viên lớn. Việc tích hợp hệ thống theo dõi dựa trên GPS làm giảm đáng kể sức người, mang lại hiệu quả cao trong việc quản lý.

Báo cáo này tôi sẽ nghiên cứu trình bày theo hướng tìm giải pháp để xây dựng một hệ thống hoàn chỉnh bao gồm thiết bị giám sát hành trình, phần mềm xử lý tại trung tâm và phần mềm giám sát trên bản đồ Google.

Với mục đích để nâng cao chất lượng quản lý và theo dõi hệ thống xe khách hiện tại. Đề tài đã đi sâu và giải quyết được các vấn đề :

-Tạo được nhiều cấp, phân quyền cho nhiều thành phần quản lý.

-Xây dựng hệ thống cho phép người sử dụng có thể truy cập xem lịch sử và thời gian thực của xe trên bản đồ.

-Tạo hệ thống với giao diện trực quan dễ sử dụng cho người dùng.

-Phân vùng và chức năng cảnh báo cho xe.

Bố cục của luận văn bao gồm phần mở đầu, phần kết luận và ba chương nội dung được tổ chức như sau:

Chương 1: Tổng quan về GPS.

Chương 2: Hệ thống GPS ứng dụng TRACCAR.

Chương 3: Chương trình quản lý xe ứng dụng công nghệ GPS TRACCAR

**CHƯƠNG 1: TỔNG QUAN VỀ GPS**

# 1.1 Một số khái niệm cơ bản GPS

## 1.1.1 Lịch sử GPS

- Hệ thống Định vị Toàn cầu (GPS) là hệ thống xác định vị trí dựa trên vị trí của các vệ tinh nhân tạo, do Bộ Quốc phòng của các nước tiên tiến như Hoa Kỳ (Mỹ), Liên Xô, trung Quốc và các nước Liên Minh Châu Âu :Pháp, Anh, Ý, Đức, Hà Lan, Ba Lan...vv thiết kế, xây dựng để vận hành và quản lý ban đầu để ứng dụng cho mục đích mục đích quân sự, rồi qua thời gian cho phép sử dụng trong dân sự .

- Ngày nay, nền kinh tế các nước phát triển kéo theo nhu cầu cuộc sống cần sử dụng thiết bị công nghệ thông minh và bùng nổ, sau này hình thành thêm thiết bị GPS cho xe tải, oto, xe khách hay người ta còn gọi là thiết bị giám sát hành trình cho Ô tô, hộp đen xe tải.

- Trong không gian đến thời điểm hiện tại gồm 27 vệ tinh (24 vệ tinh hoạt động và 3 vệ tinh dự phòng) nằm trên các quỹ đạo xoay quanh trái đất. Chúng cách mặt đất 20.200 km, bán kính quỹ đạo 26.600 km. Chúng chuyển động ổn định vá quay hai vòng quỹ đạo trong khoảng thời gian gần 24 giờ với vận tốc 7 nghìn dặm một giờ.

- Các vệ tinh GPS truyền tín hiệu từ không gian để truyền về máy thu GPS tại Mặt Đất, sử dụng những tín hiệu này để tính toán vị trí trong không gian 3 chiều dựa vào (kinh độ, vĩ độ và độ cao) tại thời gian hiện tại.Trong cùng một thời điểm, tọa độ của một điểm trên mặt đất sẽ được xác định nếu xác định được khoảng cách từ điểm đó đến ít nhất ba vệ tinh.

  -GPS hoạt động trong mọi điều kiện thời tiết và nhiệt độ không gian khác nhau và mọi nơi trên Trái Đất với đầy đủ hai mươi bốn giờ.Người sử dụng thiết bị GPS cho ô tô,xe máy, xe tải không phải tốn bất kỳ khoản nào đối với vệ tinh mà căn bản là phải tốn phí để mua thiết bị GPS giám sát hành trình và phí cài đặt thiết bị.

- Các vệ tinh bay xung quanh trái đất theo một quỹ đạo chính xác truyền tín hiệu về trái đất, từ đó GPS nhận tín hiệu, phân tích dữ liệu xác định chính xác vị trí của vật thể đã cài đặt thiết bị định vị từ trước.

## 1.1.2 Cấu trúc hệ thống GPS

GPS gồm 3 phân vùng:

* Phần không gian (space segment)
* Phần điều khiển (control segment)
* Phần người sử dụng (user segment

Mô hình ba thành phần của GPS như hình 1.1



Hình 1.1.Sơ đồ liên quan giữa ba phần của GPS.

## 1.1.3 Cấu trúc tín hiệu GPS

Một thành phần quan trọng của hệ thống GPS là tín hiệu phát từ vệ tinh đến các máy thu. Việc phát và thu tín hiệu là cơ sở cho việc đo đạc hệ thống GPS, vậy tín hiệu GPS có cấu trúc như thế nào?

Tín hiệu vệ tinh là song điện từ. Sóng điện từ được dùng cho mục đích đo đạc có những thông số đặc trưng, được nghiên cứu, thử nghiệm đảm bảo các yêu cầu nghiêm ngặt về độ chính xác, tính ổn định và yêu cần kỹ thuật khác. Về mặt vật lý, tín hiệu vệ tinh có các thông số cơ bản đó là bước sóng, tần số và các mã điều biến trên song tải.

***Bảng 1.1: Các thành phần của tín hiệu và tần số tương ứng***

|  |  |
| --- | --- |
| Thành phần | Tần số(MHz) |
| Tần số chuẩn |  |
| Sóng tải L1 | 154.f0= 1575,42Mhz (λ=19,0cm) |
| Sóng tải L2 | 120.f0 = 1227,60Mhz(λ= 24,4 cm) |
| P - Code | f0= 1,023 |
| C/A Code | f0/10 = 1,023 |
| Thông tin đạo hàng | f0/204600 = 50.10-6 |

-Tần số cơ bản của sóng truyền tín hiệu vệ tinh của hệ thống GPS là fo=10.23 MHz.

-Sóng tải có bước sóng L1 = 19cm với tần số 54\*fo = 1575,42MHz

-Sóng tải có bước sóng L2=24,4cm với tần số 120\*fo = 1227,60MHz

## 1.1.4 Các trị đo GPS

Trị đo GPS là những số liệu máy thu GPS nhận được từ tín hiệu của vệ

tinh truyền tới, mỗi vệ tinh GPS phát 4

thông số cơ bản dùng cho việc đo đạc

chia thành 2 nhóm bao gồm:

+ Nhóm trị đo Code

-C/A Code

-P Code

+ Nhóm trị đo pha:

-L1

-Carrie

-L2 Carrie

-Tổ hợp L1/L2

Các trị đo này có thể sử dụng riêng biệt hoặc kết hợp để xác định

khoảng cách đến từng vệ tinh.

Mô hình toán học của tín hiệu GPS:

Trên tần số L1 = 1575.42 MHz:

s(t) = d(t)c(t)cos(+ d(t)p(t)sin(

Trên tần số L2 = 1227.60 MHz:

s(t) = d(t)p(t)sin(

Trong đó:

d(t) : dữ liệu tần số 50bps

c(t) : mã C/A tần số 1.023MHz

p(t) : mã P tần số 10.23 MHz

ω: tần số sóng mang

# 1.2 Tình hình GPS trong nước và ngoài nước

## 1.2.1 Tình hình trong nước

Ngày nay với sự phát triển chóng mặt của công nghệ. Một số web ứng dụng GPS đã được đưa vào sử dụng. Nhằm giúp con người quản lý được vị trí hiệu quả. Một số thiết bị cẩm tay hay định vị vị trí hiện tại.

Một trong những ứng dụng phổ biến nhất của ứng dụng GPS là công ty Grap đã đưa ra app của họ cho người dùng sử dụng. Nhờ vào GPS mà các tài xế xe có thể định vị khách hàng của họ và kiểm soát vị trí của các tài xế để đưa ra những thông báo cho tài xế hợp lý.

Ứng dụng website <http://www.protrack365.com> quản lý xe tải, xe khách đã đưa GPS theo 1 hướng mới. Theo dõi hành trình chuyến đi của xe hay kiểm tra tốc độ dựa trên thời gian và quản đường xe đi được.

## 1.2.2 Tình hình ngoài nước

Một số ông lớn của công nghệ đã phát triển GPS lên một tầm cao mới. Họ đã đưa ra rất nhiều ứng dụng.

Google là một trong những công ty đi tiên phong trong lĩnh vực GPS. Họ đã làm rất nhiều ứng dụng về GPS 1 trong số đó là Google Map chế độ xem phố. Ứng dụng GPS tạo ra các địa điểm trên bản đồ hay xác định vị trí của con người trên bản đồ.

Ngoài ra nhiều nước sử dụng GPS để định vị dùng vào mục đích quân sự hoặc kiểm soát con người.

## 1.2.3 Tổng kết

GPS đã ứng dụng rất nhiều vào cuộc sống. Giúp quản lý công việc hiệu quả hơn rất nhiều. Tương lai GPS sẽ còn phát triển hơn thế nữa, nó sẽ kiểm soát mọi thứ.

Hiện tại GPS chưa quản lý được bản số xe, ngày hết hạn thiết bị GPS vậy nên ứng dụng này sẽ giải quyết vấn đề trên.

# CHƯƠNG 2: HỆ THỐNG GPS VÀ ỨNG DỤNG TRACCAR.

# 2.1 Xây dựng hệ thống GPS

## 2.1.1 Thiết bị GPS GT02:

2.1.1.1 **Thông số kỹ thuật**

**\* Mạng GSM:** SIM900B

– Tần số: 850/ 900/ 1800/ 1900 MHz

– GPRS: GPRS multi-slot class 10/8

– GPRS mobile station class B

### 2.1.1.2 **Trạng thái thiết bị làm việc**

**-Đèn trạng thái (Nguồn/trạng thái làm việc)**

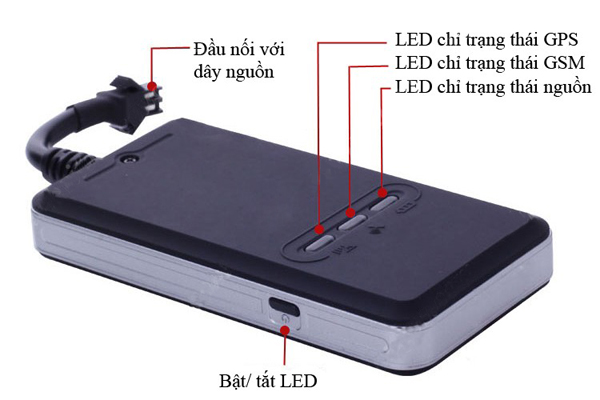
|  |  |
| --- | --- |
| Trạng thái đèn | Mô tả |
| Nhấp nháy | Chỉ cung cấp năng lượng, hoạt động bình thường |
| Đèn sáng liên tục | Không hoạt động bình thường |
| Đèn không sáng | Tắt/lỗi |

**-Đèn xanh da trời (trạng thái GPS)**

|  |  |
| --- | --- |
| Trạng thái đèn | Mô tả |
| Nhấp nháy (Khoảng 1s) | Tìm kiếm tín hiệu GPS |
| Đèn sáng liên tục | GPS xác định được vị trí |
| Đèn không sáng | GPS không xác định được  vị trí |

**-Đèn Xanh lá (chỉ báo tình trạng GSM)**

|  |  |
| --- | --- |
| Trạng thái đèn | Mô tả |
| Nhấp nháy nhanh | Khởi động GPS |
| Đèn sáng liên tục | GSM hoạt động / bắt đầu GPRS |
| Đèn không sáng | Không có tín hiệu GSM |



### 2.1.1.3 **Cài đặt thẻ SIM:**

**-Mỗi GT02 cần lắp thẻ SIM GSM.**

**-Thẻ SIM được sử dụng phải được kích hoạt GPRS.**

- Lắp sim vào điện thoại kiểm tra để đảm bảo SIM đó có thể gửi và nhận tin nhắn và GPRS đã được kích hoạt.

- Kiểm tra xem phí thẻ SIM của bạn quá hạn.

# 2.2 Giới thiệu hệ thống server TRACCAR:

## 2.2.1**Cài đặt hệ thống:**

**- Chuẩn bị IDE Netbean.**

**- Cài đặt JDK 7 trở lên.**

**=> Tiến hành cài đặt.**

**- Sau khi cài đặt vào đường dẫn** <http://localhost:8082> **để xem kết quả.**

## 2.2.2 **Sơ đồ hoạt động của server:**

**Clinet**

**(Website)**

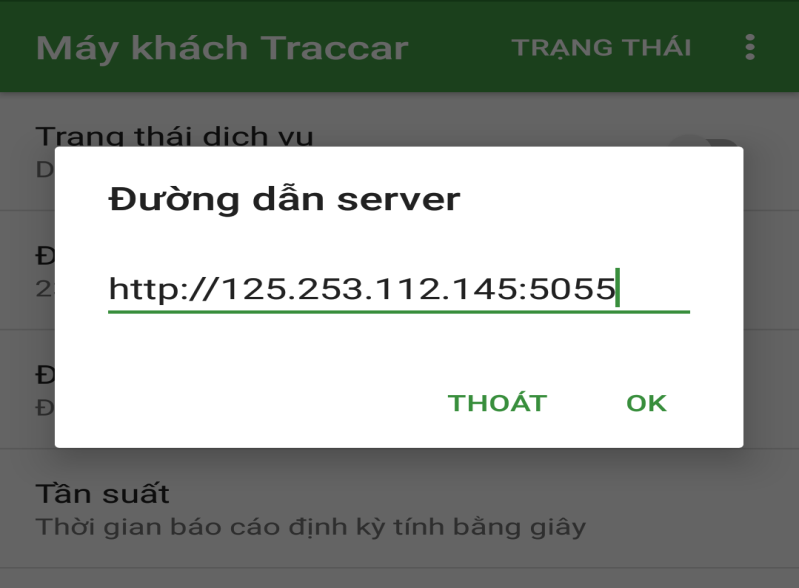
**Server**

**Database**

**Thiết bị**

## 2.2.3 **Thiết bị kết nối với server:**

**- Thiết bị phải cài đặt cấu hình của server cần kết nối.**



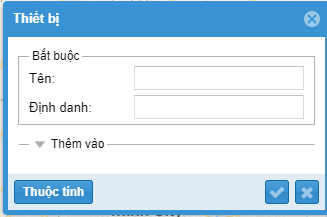
* <Http://125.253.112.145> là địa chỉ server cần cài đặt thiết bị.
* 5055 là port của thiết bị.(*chi tiết các port tham khảo ở* [*https://www.traccar.org/devices*](https://www.traccar.org/devices)*)*
* Sau đó bật thiết bị lên.(kiểm tra chắc chắn rằng thiết bị đã bật GPS)

## 2.2.4 **Server thêm thiết bị:**

**- Truy cập vào đường dẫn** <http://localhost:8082> **để thêm thiết bị.**



* **Bên góc trái màn hình sẽ hiện ra 1 danh sách các thiết bị đã được thêm bởi người dùng đăng nhập trước đó.**
* Muốn thêm thiết bị ta bấm vào dấu cộng trên đầu danh sách.



* Người dùng cấp 1 cái tên tùy ý để quản lý thiết bị.
* Định danh : Là mã của thiết bị đó.(mỗi thiết bị có 1 mã khác nhau)

## 2.2.5 **Cơ chế hoạt động của server:**

**Gửi tín Gửi tín hiệu Hiển thị**

Website

Server

(**Sử lý)**

* Thiết bị sẽ gửi tín hiệu bao gồm:
  + Vĩ độ
  + Kinh độ
  + Độ cao
  + Tốc độ
  + Mức pin
  + Chuyển động
  + Hướng
* Tùy vào các thiết bị khác nhau sẽ gửi thêm 1 số thuộc tính khác nhau.
* Server sẽ tính toán và sử lý cho ra địa điểm và hiển thị chúng trên Google Map.
* Sau đó Server sẽ lưu chúng vào Database.

## 2.2.6 **Hệ thống thông báo:**

- Loại cảnh báo:

* Gửi từ web
* Gửi qua mail
* Gửi qua tin nhắn
* Các trường hợp cảnh báo:
  + Thiết bị đang di chuyển
  + Thiết bị ngoại tuyến
  + Thiết bị trực tuyến
  + Thiết bị vượt quá tốc độ
  + Thiết bị ra ngoài giới hạn địa lý
  + Ngắt khóa
  + Mở khóa
* Cần phải cấu hình thiết bị để được cảnh báo theo dạng.

## 2.2.7 **Thiết lập Cơ sở dữ liệu:**

### 2.2.7.1 **Thay đổi cơ sở dữ liệu:**

**- Mặt định khi cài đặt sẽ lấy cơ sở dữ liệu của oracle.**

**- Chuyển đổi cơ sở dữ liệu qua sql server.**

**- Thay đổi trong file** default

<entry key='database.driver'>com.microsoft.sqlserver.jdbc.SQLServerDriver</entry>

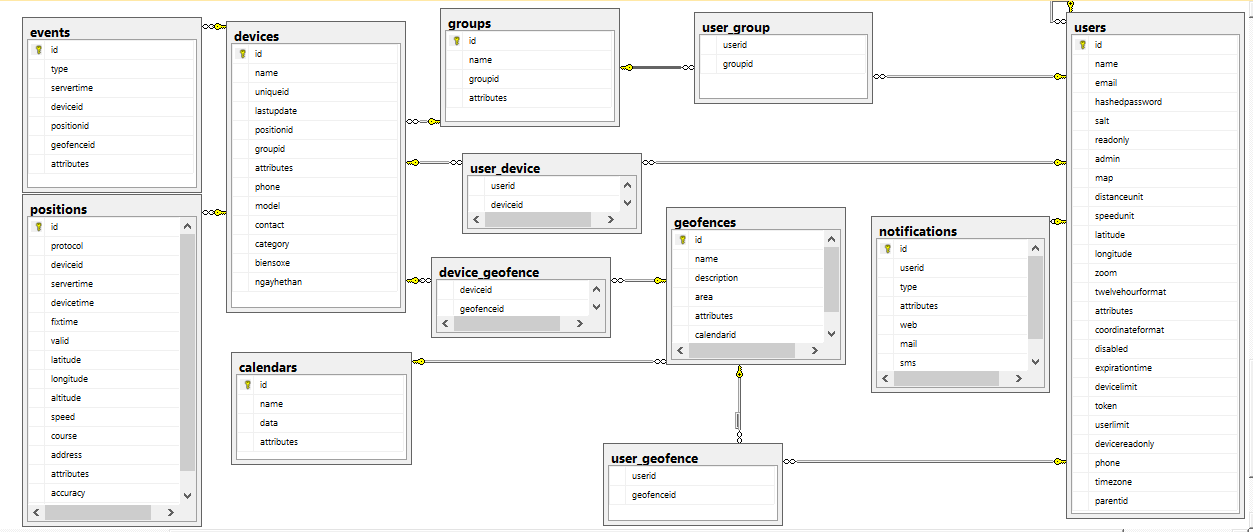
<entry key='database.url'>jdbc:sqlserver://[serverName][\instanceName];user=[userName];password=[password];databaseName=[database];

</entry>

<entry key='database.user'>[userName]</entry>

<entry key='database.password'>[password]</entry>

* [serverName] – Địa chỉ của server
* [\instanceName] tên server
* [userName] – tên đăng nhập sql server
* [password] – mật khẩu sql server
* [database] – tên database
  + - 1. **Cơ sở dữ liệu.**



|  |  |  |
| --- | --- | --- |
| **STT** | **Tên bảng** | **Chức năng** |
| **1** | **Users** | **Đây là bảng chính lưu trữ thông tin cơ bản của người dùng.**  **id : số thứ tự đăng ký của người dùng.**  **name : tên người dùng.**  **Email : tên đăng nhập**  **Hashedpassword : mật khẩu**  **Readonly : quyền hạn chỉ đọc**  **Admin : có phải là admin**  **Map : kiểu bản đồ thể hiện trên website**  **Distanceunit : đơn vị khoản cách tính trên bản đồ**  **Speedunit : đơn vị tốc độ tính trên bản đồ**  **Latitude : vĩ độ**  **Longitude : kinh độ**  **Zoom : đơn vị phóng to khi lần đầu tiên mở bản đồ**  **Twelvehourformat : định dạng 12 giờ**  **Attributes : thuộc tính thêm của user**  **Coordinateformat :cấu trúc tọa độ**  **Disabled : Kích hoạt user**  **Expirationtime : thời gian hết hạn của user**  **Devicelimit : giới hạn thiết bị**  **Token : mã tosken để đăng nhập dựa trên token**  **Userlimit : giới hạn số lượng người dùng**  **Devicereadonly : quyền hạn chỉ xem thiết bị**  **Phone : số điện thoại user**  **Timezone : múi giờ**  **Parentid : id phân biệt giữa mỗi cấp trên quản lý mình** |
| **2** | **devices** | **Đây là bản lưu trữ thông tin thiết bị**  **Id : số thứ tự đăng ký thiết bị**  **Name : tên thiết bị**  **Uniqueid : mã định danh thiết bị**  **Lastupdate : lần cuối cùng cập nhật thiết bị**  **Positionid : mã liên kết với vị trí thiết bị**  **Groupid : mã liên kiếm với nhóm thiết bị**  **Attributes : thuộc tính thêm của thiết bị**  **Phone : số điện thoại thiết bị đó**  **Model : kiểu mẫu**  **Contact : thông tin liên hệ**  **Category : phân loại thiết bị**  **Biensoxe : biển số xe của thiết bị đó**  **Ngayhethan : hạn sử dụng của thiết bị** |
| **3** | **positions** | **Đây là bản lưu trữ vị trí của thiết bị dựa trên mã id thiết bị**  **Id : số thứ tự của bản vị trí**  **Protocol : giao thức của thiết bị**  **Deviceid : mã thiết bị đã gửi vị trí**  **Servertime : thời gian hiện tại của server**  **Devicetime : thiết bị gởi lên**  **Fixtime : khoảng delay thời gian nhận được của server**  **Valid : vị trí không còn xác định**  **Latitude : vĩ độ**  **Longitude : kinh độ**  **Altitude : độ cao**  **Speed : tốc độ**  **Course : phương hướng**  **Address : địa chỉ trên bản đồ**  **Accuracy : tính chính xác xác định trên bản đồ**  **Network : địa chị lớp mạng.** |
| **4** | **notifications** | **Đây là bảng cài đặt và lưu trữ thông báo cho user**  **Id : số thứ tự trong bảng**  **Userid : mã liên kết với bản users**  **Type : nội dung thông báo theo kiểu**  **Web : thông báo qua website**  **Mail : thông báo qua mail người dùng cung cấp**  **Sms : thông báo qua số điện thoại** |
| **5** | **Events** | **Đây là bản lưu trữ nội dung dự kiện của thiết bị**  **Id : thứ tự trong bảng**  **Type : nội dung lưu trữ theo kiểu**  **Servertime : thời gian server xác nhận**  **Deviceid : mã thiết bị sảy ra sự kiện**  **Positionid : mã vị trí sảy ra sự kiện**  **Attributes : trạng thái sự kiện đó** |
| **6** | **groups** | **Đây là bản lưu trữ các nhóm thiết bị**  **Id : thứ tự trong bảng**  **Name : tên nhóm**  **Groupid : id của nhóm**  **Attributes : trạng thái nhóm** |
| **7** | **geofences** | **Đây là bản lưu trữ các vùng giới hạn**  **Id : thứ tự trong bảng**  **Name : tên vùng**  **Description : thông tin về vùng đó**  **Area : bảng đồ khu vực**  **Attributes : trạng thái hoạt động của vùng đó** |
| **8** | **calendars** | **Đây là bảng lưu lại lịch trình xác lập các vùng**  **Id : thứ tự trong bảng**  **Name : tên mốc thời gian xác định**  **Data : thời gian xác lập vùng**  **Attributes : trạng thái** |

* **Các bản liên kết:**

**+ user\_group : Các user sẽ thuộc các nhóm , user có thể tham gia nhiều nhóm khác nhau. Người dùng sẽ thuận tiên quản lý thiết bị theo nhóm.**

**+ user\_device : User sẽ có các thiết bị khác nhau.họ sẽ quản lý thiết bị của họ. Tùy thuộc vào quyền hạn của họ mà cách quản lý khác nhau.**

**+ user\_geofence : User sẽ quản lý các vùng mà họ tạo ra. Họ có thể chỉnh sửa các vùng phù hợp với vùng di chuyển xe của họ muốn. Thông qua đó họ có thể cảnh báo thiết bị khi đi vào vùng hoặc ra khỏi vùng đã được xác lập bởi họ.**

**+ device\_geofence : Thiết bị thuộc vùng mà User thêm vào. Mỗi thiết bị sẽ được thêm vào các vùng được xác định hoặc không.**

* **Chú ý: Liên kết Users với Users : là là liên kết giữa các user trong cùng một bảng tạo sự ràng buộc với nhau thông qua thuộc tính parenID liên quan tới cấp quyền cho users và phân cấp của user.**

2.2.8 **Biểu đồ phân rã chức năng.**

Hệ thống quản lý

Cấu hình server

Thống kê thiết bị

Thống kê người dùng

Quản lý người dùng

Đăng ký thiết bị

Đăng ký người dùng

Quản lý thiết bị

Xem báo cáo thiết bị

Xem vị trí thiết bị

Báo lỗi

Quản lý thông tin cá nhân

Xe lại thiết bị

Đăng nhập

Hệ thống quản lý

Chức năng người dùng

# CHƯƠNG 3: CHƯƠNG TRÌNH QUẢN LÝ XE

# 3.1 Giới thiệu chương trình quản lý xe.

**Ngày nay xe là 1 phương tiện không thể thiếu của mỗi gia đình. Phương tiện di chuyển tăng 1 cách chóng mặt trong những năm qua. Qua đó quản lý xe là điều không thể thiếu cho 1 đất nước nhiều xe. Thông qua đó ta có thể kiểm soát được quá trình di chuyển của 1 lượng lớn xe.**

**Chương trình quản lý xe là 1 phần mềm được xây dựng dựa trên mã nguồn mở TracCar. Chương trình tạo ra nhằm mục đích quản lý xe của công ty, xe nhà riêng. Giúp công việc trở nên hiệu quả, công ty có thể quản lý tất các xe của mình tránh tình trạng thất thoát công việc. Hay là muốn xác định vị trí xe nhằm tạo điều kiện phát triển xã hội.**

**Tổng kết: Biển số xe và hệ thống người sử dụng xe là 1 vấn đề cần quản lý.**

**Từ hệ thống cơ bản ban đầu, để xây dựng thành 1 ứng dụng hoàn thiện tác giả còn phải bổ sung nhiều chỉnh sửa bao gồm**

|  |  |  |
| --- | --- | --- |
| STT | Chức năng | Mô tả |
| 1 | Thay đổi giao diện Kinhtech | Giao diện người dùng cần được thay đổi trực quan. |
| 2 | Thêm biển số xe | Thêm 1 thuộc tính thông tin vào thiết bị. Quản lý thông tin thiết bị dựa trên biển số xe. |
| 3 | Xem lại trực quan | Hệ thống chỉ lưu trữ vị trí địa điểm vào database. Cần thể hiện xem lại trên google Map để dễ xem lại. Cùng với Animation chúng ta có thể dễ nhận ra quảng đường mà xe di chuyển được trong khoảng thời gian xác định |
| 4 | Phân quyền hệ thống | Hệ thống cần được phân chia thành nhiều cấp bậc. Dễ dàng cho việc quản lý. Trong hệ thống hiện tại chỉ chia được làm 3 lớp nhưng mọi người trong hệ thống có thể nhìn thấy nhau và tương tác qua lại. Vì thế cần phải tạo ra sự tối ưu cần thiết về quản lý người dùng. |
| 5 | Ngày hết hạn của thiết bị | Mỗi thiết bị có ngày hết hạn thiết bị. Vì thế cần phải thêm vào thuộc tính ngày hết hạn để quản lý kiểm soát ngày hết hạn của thiết bị. Đồng thời sẽ thông báo ngày hết hạn khi thiết bị đã hết hạn. |
| 6 | Hiển thị thông tin trên bản đồ | Việc quan sát xe sẽ thuận tiện hơn khi ta kích vào xe trên bản đồ sẽ hiện ra thông tin xe. Cần ra map maker gắn với từng xe để thể hiện thông tin xe. |
| 7 | Hiển thị nhiều loại map | Tạo ra nhiều loại map khác nhau tùy vào nhu cầu dùng của mỗi người. Người dùng có thể quan sát thiết bị của mình trên nhiều giao diện map. |
| 8 | Loại bỏ 1 số khung không cần thiết | Cần để cho giao diện dễ nhìn. Thuận tiện cho việc quản lý. Loại bỏ 1 số thứ dư thừa trên giao diện. Tinh chỉnh giao diện phù hợp với người sử dụng . Ta cần gọt bỏ 1 số khung không cần thiết trên giao diện |

# 3.2 Chức năng chương trình.

## 3.2.1 ****Phân quyền user:****

**Admin**

**Đại lý**

**Nhà phân phối**

**Người dùng**

* **Gồm 4 cấp người dùng.**

**- Admin**

**- Đại lý**

**- Nhà phân phối**

**- Người dùng**

**Admin: sẽ nắm tất cả mọi quyền.bao gồm quyền thêm thiết bị với kiểm soát tất cả các đại lý.**

**Đại lý: sẽ chịu trách nhiệm tại ra các nhà phân phối của họ.họ sẽ cung cấp thiết bị của họ và kiểm soát Nhà phân phối.đồng thời kiểm soát quyền hạn của Nhà phân phối.**

**Nhà phân phối: nhiệm vụ sẽ add thiết bị của họ và giao cho các người dùng sử dụng chúng.Thiết lập ngày hết hạn thiết bị…**

**Người dùng: Sử dụng thiết bị.**

* **Xây dựng phân quyền.**

Để xây dựng chức năng phân quyền user ta cần thêm 1 thuộc tính vào user (parentId). Thuộc tính này giúp mọi user sẽ tách biệt với nhau.Mỗi user sẽ có 1 prentId của cấp trên của mình vì thế cấp trên sẽ chỉ nhìn thấy user của họ mà không thấy được user của người khác. Tránh tình trạng quản lý người dùng không được hiệu quả tạo ra sự lẫn lộn giữa các người dùng.

**Quản lý 2**

**Quản lý 1**

**Người dùng 2**

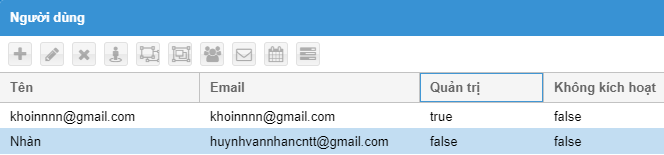
**Người dùng 1**

**-** Mỗi user chỉ được thêm 1 parentId và đó cũng là id cấp trên của mình mà trước đó quản lý đã tạo ra user này.

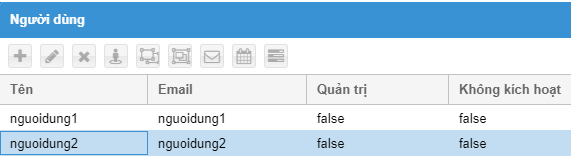
-**Người dùng 2** sẽ không được **Quản lý 1** quản lý nếu nếu **Quản lý 1** không tạo ra **người dùng 2**. Nó có nghĩa là Quản lý chỉ được quản lý user mà họ đã tạo ra.

-Với cơ chế trên ta có thể tạo ra rất nhiều cấp quản lý khác nhau mà không chỉ dừng lại là 4 cấp. Để tạo ra điều kiện quản lý chặt chẽ hơn về người dùng.

Kết quả khi đăng nhập admin chỉ nhìn thấy những user mình tạo ra.



Mà không nhìn thấy được 2 người dùng mà user Nhàn đã tạo.



## 3.2.2 ****Thiết kế giao diện chương trình.****

### 3.2.2.1 ****Giới thiệu framework Ext JS****

**Giao diện chương trình được thiết kế được bởi framework Ext JS. Là 1 framework được xây dựng trên ngôn ngữ JavaScript ,ngôn ngữ cực mạnh trong việc phát triển giao diện hiện nay.**

**ExtJS là viết tắt của JavaScript Extended. Đó là một khuôn khổ JavaScript và một sản phẩm của Sencha, dựa trên YUI (Yahoo! User Interface). Nó cơ bản là một nền tảng phát triển ứng dụng desktop với giao diện người dùng hiện đại.**

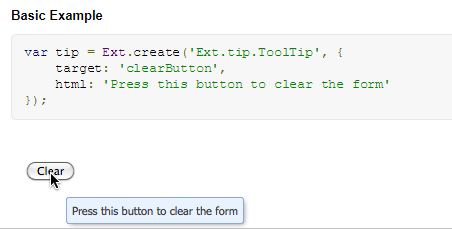
**Ext JS là cơ bản được sử dụng để tạo các ứng dụng máy tính để bàn. Nó hỗ trợ tất cả các trình duyệt hiện đại như IE6 +, FF, Chrome, Safari 6+, Opera 12+, vv Trong khi đó, một sản phẩm của Sencha, Sencha Touch được sử dụng cho các ứng dụng di động.**

**Ext JS dựa trên kiến ​​trúc MVC / MVVM. Phiên bản mới nhất của Ext JS 6 là một nền tảng duy nhất, có thể được sử dụng cho cả máy tính để bàn và ứng dụng di động mà không cần phải đoạn mã khác nhau cho nền tảng khác nhau.**

### 3.2.2.2 ****Component trong Ext JS****

**Thiết kết giao diện bây giờ không còn nằm ở việc thiết kế bằng HTML và CSS nữa mà thay vào đó famework đã thiết kế sẵn các component để thuận tiện cho việc xây dựng giao diện. Với các thuộc tính ta có thể dễ dàng tùy chỉnh vị trí cũng như là màu sắc đến hình dạng… của component đó.**

**Đây là ví dụ cho 1 Tootip được thiết kế sẳn trong famework.**



**Ta có thể chỉnh sửa them cho phù hợp với người dùng. Them của famework của được thiết kế theo mẫu chung vì thế chúng ta chỉ cần tải them về rồi gắn vào website của mình nó sẽ hiện lên them mà bạn đã cài.**

addScriptFile('//cdnjs.cloudflare.com/ajax/libs/extjs/' + extjsVersion + '/classic/locale/locale-' + locale.languages[locale.language].code + '.js');

addStyleFile('https://cdnjs.cloudflare.com/ajax/libs/extjs/' + extjsVersion + '/classic/theme-neptune/resources/theme-neptune-all.css');

addScriptFile('//cdnjs.cloudflare.com/ajax/libs/extjs/' + extjsVersion + '/classic/theme-triton/theme-triton.js');

addStyleFile('//cdnjs.cloudflare.com/ajax/libs/extjs/' + extjsVersion + '/packages/charts/classic/triton/resources/charts-all.css');

addStyleFile('//cdnjs.cloudflare.com/ajax/libs/font-awesome/' + fontAwesomeVersion + '/css/font-awesome.min.css');

addStyleFile('//cdnjs.cloudflare.com/ajax/libs/ol3/' + olVersion + '/ol.css');

**Ngoài ra ta cũng có thể chỉnh sửa đường dẫn để trực tiếp lấy file them online qua nơi lưu trữ.**

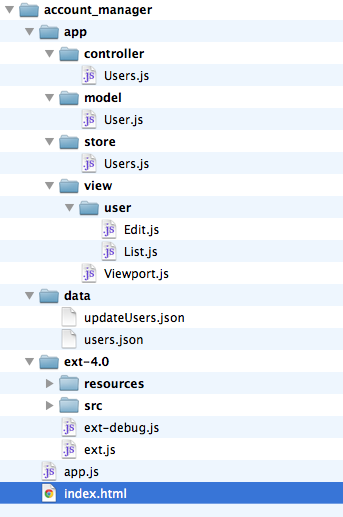
**Chú ý: ta cũng có thể CSS cho các component của famework bằng cách tải them mà ta muốn custommer về rồi vào file style.css để thực hiện thay đổi hay chỉnh sửa 1 số thuộc tính ta muốn.**

### 3.2.2.4 Kiến trúc MVC

**Có rất nhiều kiến ​​trúc MVC, hầu hết trong số đó là một chút khác biệt so với một số khác. Đây là cách chúng ta định nghĩa chúng ta:**

* ***Model* là một tập hợp các lĩnh vực và dữ liệu của họ (ví dụ như một mô hình tài khoản với username và password lĩnh vực). Mô hình biết làm thế nào để tồn tại bản thân thông qua các gói dữ liệu, và có thể được liên kết với các mô hình khác thông qua các hiệp hội. Mô hình làm việc rất nhiều như Ext JS 3 Ghi lớp, và thường được sử dụng với**[**cửa hàng**](http://www.objis.com/formationextjs/lib/extjs-4.0.0/docs/api/Ext.data.Store.html)**để trình bày dữ liệu vào lưới và các thành phần khác**
* ***View* là bất kỳ loại thành phần - lưới, cây cối và các tấm đều xem.**
* ***Controllers*  là nơi đặc biệt để đưa tất cả các mã mà làm cho việc ứng dụng của bạn - cho dù đó là quan điểm render, Models instantiating, hoặc bất kỳ logic ứng dụng khác.**

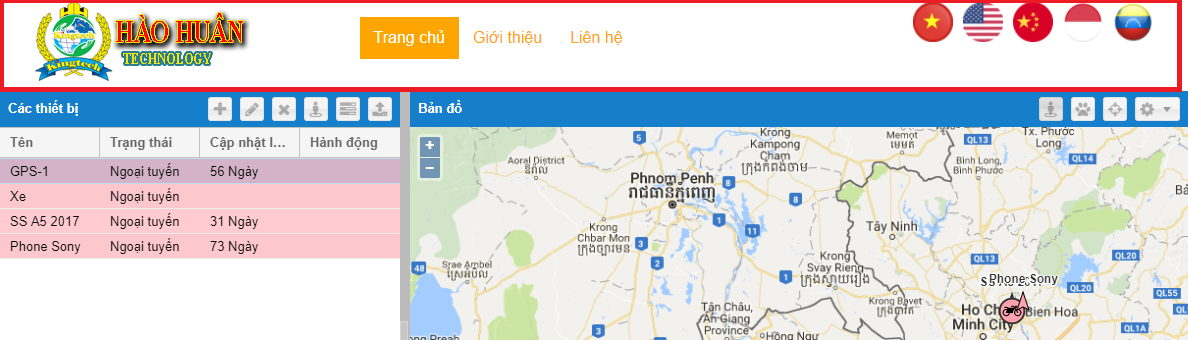
**Cấu trúc tập tin được tổ chức như sau qua ví dụ sau đây :**



### 3.2.2.5 ****Phát triển giao diện****

* **Xây dựng hearder trang.**

**Header được xây dựng bằng Angular js, được cấu hình 3 taskbar (Trang chủ, giới thiệu, liên hệ)**



**Từng taskbar ta sẽ điều hướng về mỗi trang khác nhau để thể hiện nội dung, trang chủ là mặc định.**

* **Trang chủ: Gồm thông tin thiết bị xe và hệ thống quản lý xe. Nơi người dùng có thể xem trực tiếp thiết bị của mình.**
* **Giới thiệu : Gồm những thông tin về công ty cùng với những thông tin ứng dụng giúp người sử dụng có thể hiểu thêm về hệ thống.**
* **Liên hệ : Bao gồm thông tin liên hệ của công ty. Để khi hệ thống gặp lỗi hoặc có thắc mắc về chương trình ta có thể liên hệ cho công ty**
* **Hiển thị map marker hiện thông tin xe**

**Thông tin xe sẽ được hiển thị ra cho người sử dụng xem. Họ sẽ biết được Tên xe cũng như ngày hết hạn của xe và nhiều thông tin khác. Giúp họ có thể xem được thông tin của xe 1 cách nhanh nhất có thể mà không cần phải tìm kiếm xe của mình 1 cách vất vã.**



**Map marker là 1 trong những tiện ích của google map. Map marker sẽ hiển thị ra thông tin đối tượng được hướng đến.Trước hết cần get key Google map để chạy api của google map.**

## 3.2.3 ****Quản lý biển số xe.****

**Trong mã nguồn mở chưa quản lý được biển số xe của từng xe. Trong khi đó bản số xe là thuộc tính để tách từng xe riêng biệt. Bản số xe của mỗi xe sẽ khác nhau, dựa vào điều này đối tượng biển số xe cần được quản lý.**

### 3.2.3.1 ****Tìm hiểu cách thức hoạt động của chương trình****

**Để xây dựng chức năng này trước hết cần phải tìm hiểu luồng và dòng dữ liệu chạy như thế nào trong chương trình. Qua đó ta có thể chỉnh sửa lại theo những gì ta muốn hoặc thêm vào những thứ chưa có trong ứng dụng quản lý mà ta xây dựng.**

**Luồng và cách thức hoạt động của chương trình như sau :**

Database

Server

Controller

DeviceManager.java

Model

(Device.java)

Website

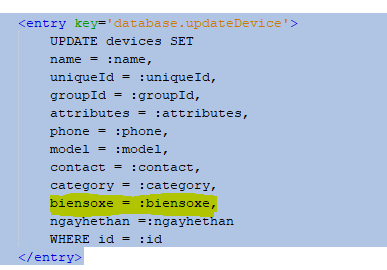
Thêm bản số xe

### 3.2.3.1 ****Thêm thuộc tính biển số xe vào database****

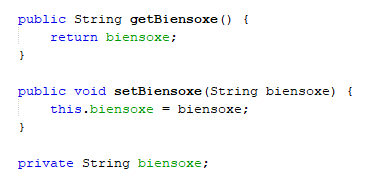
Ta cần thêm thuộc tính biển số xe vào database để lưu lại biển số các xe. Thuận tiện cho việc quản lý. Qua đó cần chỉnh 1 vài câu lệnh truy vấn cho phù hợp với chương trình.

### 3.2.3.2 ****Truy vấn dữ liệu và hiển thị liên giao diện****

* Thay đổi file default để truy vấn thêm thuộc tính biensoxe



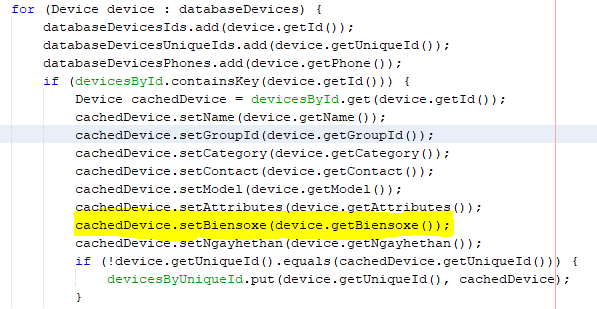
* Vào file Device.java thêm thuộc tính biển số xe vào class để đối tượng có thể nhận từ database lên và update ngược lại về database



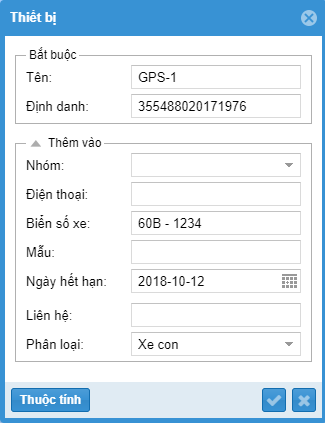
* Cấu hình để hiển thị biển số xe trên giao diện trong file Devices.js



* Sau đó vào file DeviceManager.java thêm 1 số cấu hình để kết nối giữa model vào view



* Kết quả ta được giao diện như sau

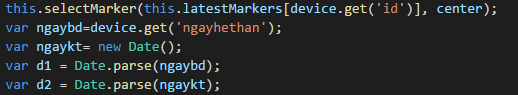


## 3.2.4 ****Thông báo ngày hết hạn thiết bị.****

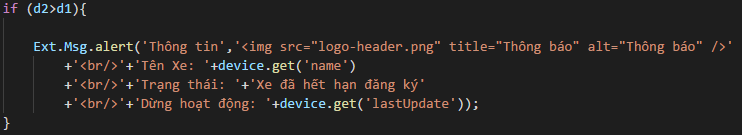
**Số lượng thiết bị GPS gắn trên xe sẽ có thời gian hết hạn. Vì thế cần thêm vào quản lý ngày hết hạn của thiết bị. Giống như biển số xe cũng cần thêm thuộc tính vào database để quả lý ngày hết hạn.**

**Để quản lý ngày hết hạn cần viết thêm 1 số đoạn mã trong chương trình.**

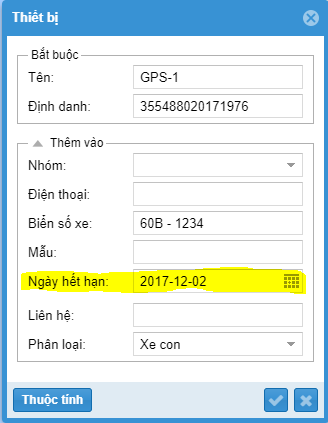
* **Ta cần lấy được ngày hiện tại và ngày thiết bị hết hạn**



* **Sau đó so sánh với nhau nếu ngày hết hạn đã vượt qua ngày hiện tại thì ta thông báo cho người dùng biết thiết bị này đã hết hạn.**



* **Gia hạn ngày hết hạn hoặc có thể kết thúc ngày hết hạn kết thúc nhanh hơn.**



* **Kết quả sau khi đã quản lý được ngày hết hạn của thiết bị sẽ thông báo trên màng hình như sau:**



## 3.2.5 ****Chế độ xem lại.****

### 3.2.5.1 Giới thiệu

**Trong khoản thời gian xác định, người sử dụng cần biết xe mình đã đi đâu dừng ở đâu hay là quản đường mà xe của mình đi được bao xa. Họ muốn nắm hết mọi thông tin của xe ở 1 khoảng thời gian mà xe đã đi qua. Chế độ xem lại này sẽ giúp họ có thể cảm nhận được xe họ đã làm gì trong khoảng thời gian đó. Cho biết tất cả mọi vị trí mà xe họ đã đi qua đồng thời còn biết được khoảng thời gian nghĩ của xe khi gặp đèn đỏ hoặc quá trình xe không di chuyển. Ta có thể điều khiển cho chế độ nhanh chậm tùy thuộc vào người theo dõi và quản đường dài hay ngắn để có cái nhìn tổng thể về quảng đường xe đã đi. Traccar đã hỗ trợ cho chúng ta API server để qua đó ta có thể truy xuất dữ liệu.**

### 3.2.5.2 Tìm hiểu API và cách request api



API là viết tắt của Application Programming Interface, phương thức kết nối với các thư viện và ứng dụng khác. Windows có nhiều API, và Twitter cũng có web API, tuy chúng thực hiện các chức năng khác nhau, với mục tiêu khác nhau.Nhìn chung, RESTful API là những API đi theo cấu trúc REST.

Cấu trúc REST là gì?

* Sự nhất quán trong cả API
* [Tồn tại không trang thái](http://stackoverflow.com/a/3105337/477958) (ví dụ, không có server-side session)
* Sử dụng HTTP status code khi cần thiết
* Sử dụng URL endpoint với logical hierarchy
* Versioning trong URL chứ không phải trong HTTP header

**Mô hình cấu trúc** REST



Nhìn chung, có bốn lệnh dùng để truy cập RESTful API:

1. **GET** để truy vấn object
2. **POST** để tạo object mới
3. **PUT** để sửa đổi hoặc thay thế một object
4. **DELETE** để loại bỏ một object

### 3.2.5.3 Requestr API TracCar

Để requestr API TracCar chúng ta cần đăng nhập vào hệ thống để lấy log user. Khi đó user mới có quyền Get các API. Tùy vào quyền user mà kết quả Get ra data khác nhau hay là quyền get API đó. Admin là quyền cao nhất có thể get mọi API.

**Để thực hiện chức năng xem lại chúng ta cần Get API từ TracCar qua api server** <http://localhost:8082/api/reports/route> **với các tham số truyền vào.**

+ deviceId: id thiết bị truyền vào.

+ from : từ ngày mà bao nhiêu

+ to : đến ngày bao nhiêu.

**Để sử dụng được API trên ta cần setup cái tham số truyền vào sử dụng phương thức GET để lấy dữ liệu về.**

$scope.Review = function(){

if($('#btnReview').hasClass('btn-warning')){

var dateBegin = moment($scope.DateBegin, "DD-MM-YYYY HH:mm:ss").format("YYYY-MM-DDTHH:mm:ss");

var dateEnd = moment($scope.DateEnd, "DD-MM-YYYY HH:mm:ss").format("YYYY-MM-DDTHH:mm:ss");

var data = '?deviceId=' + parseInt($scope.deviceId) + '&from=' + dateBegin + '&to='+dateEnd;

MyFactory.GET(MyFactory.HOST\_NAME + MyFactory.API.REPORT.ROUTE + data).then(function(response){

if(response){

$scope.RESULT = response;

console.log($scope.RESULT);

if($scope.RESULT.length > 0){

$('#btnReview').removeClass('btn-warning').addClass('btn-default');

$('#btnReview').html('Dừng');

getListPosition(response);

}

else {

$('#btnReview').removeClass('btn-default').addClass('btn-warning');

$('#btnReview').html('Xem lại');

$interval.cancel($scope.interval);

}

}

})

}

if($('#btnReview').hasClass('btn-default')){

$('#btnReview').removeClass('btn-default').addClass('btn-warning');

$('#btnReview').html('Xem lại');

$interval.cancel($scope.interval);

}

}

**Dùng javascript để đổ dữ liệu lên map.**

var tick = function(){

if($scope.LIST\_POSITION.length > 0 && count < $scope.countList){

var item = $scope.RESULT[count];

var pos1 = new google.maps.LatLng($scope.item.latitude,$scope.item.longitude);

var pos2 = new google.maps.LatLng(item.latitude,item.longitude);

var angle = google.maps.geometry.spherical.computeHeading(pos1, pos2);

var from = new Date($scope.RESULT[0].fixTime);

var to = new Date(item.fixTime);

var timeDiff = Math.abs(from.getTime() - to.getTime());

$scope.item = item;

$scope.item.totalTime = GetDuration(timeDiff);

if(angle != 0)

$scope.angle = angle;

count++;

}

if(count == $scope.countList-1){

var from = new Date($scope.RESULT[0].fixTime);

var to = new Date($scope.RESULT[$scope.countList-1].fixTime);

var timeDiff = Math.abs(from.getTime() - to.getTime());

$scope.dataEnd = {

from: $scope.RESULT[0].fixTime,

to: $scope.RESULT[$scope.countList-1].fixTime,

totalDistance: $scope.RESULT[$scope.countList-1].attributes.totalDistance,

totalTime: GetDuration(timeDiff)

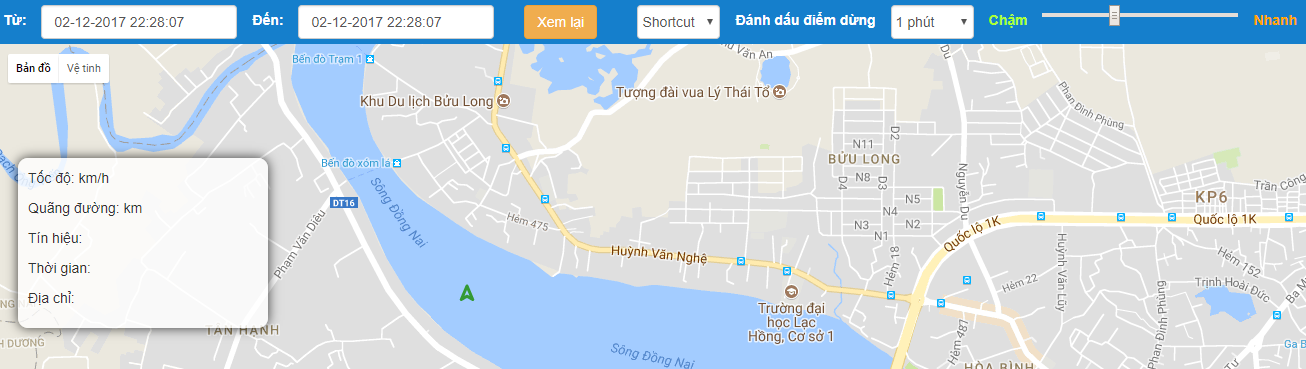
}

$('.modal').modal('show');

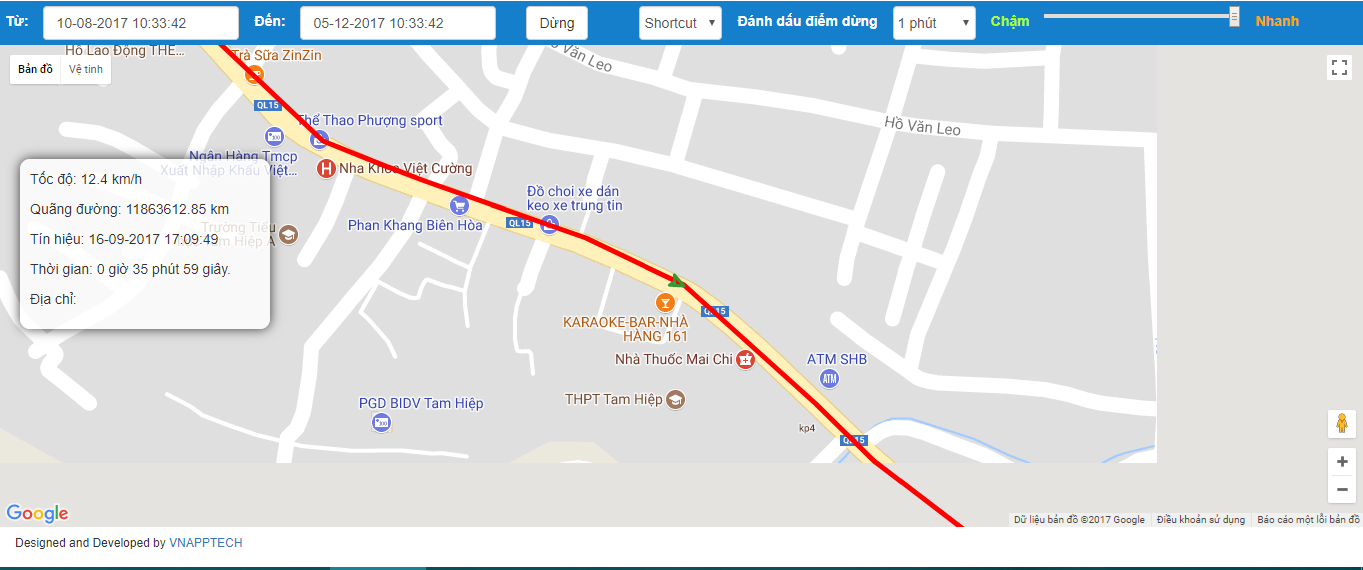
}

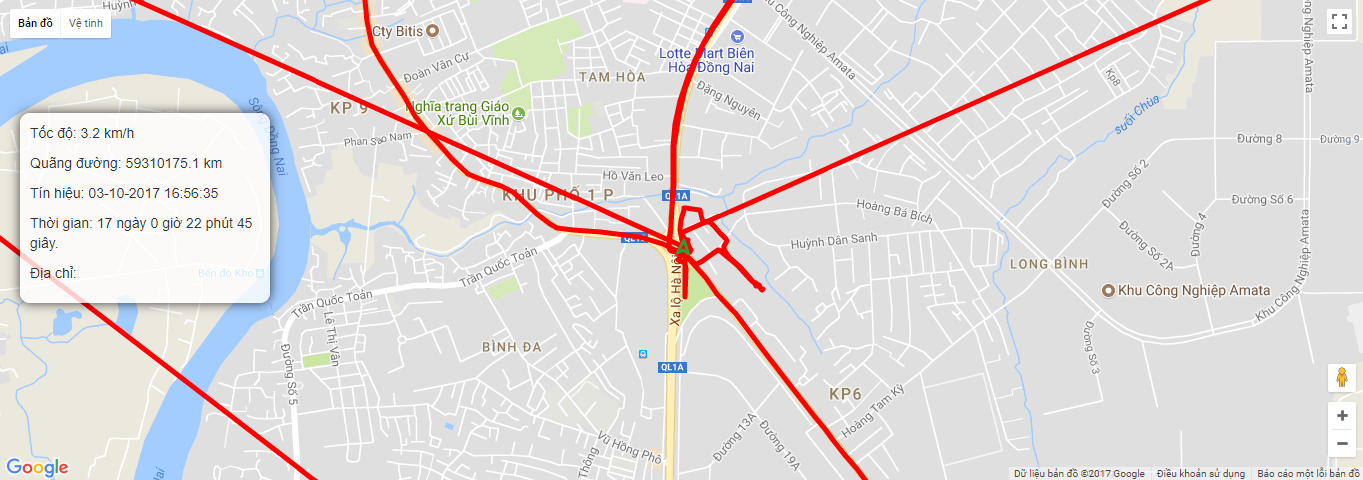
}

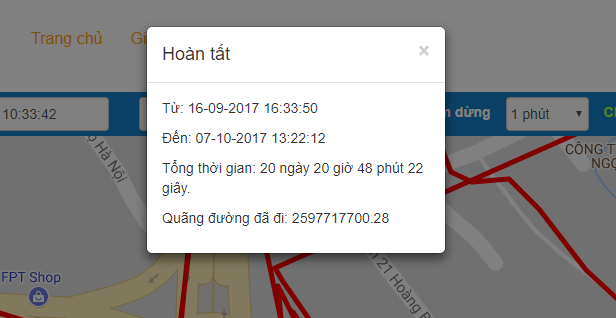
**Giao diện chức năng xem lại.**



* **Chọn khoảng thời gian cần xem lại.**
* **Shortcut chức xem lại nhanh**
* **Chức năng báo khi dừng lại.**
* **Xem với chế độ nhanh chậm tùy theo mức cần chỉnh.**
* **Sau khi cài đặt xong chọn xem lại để xem quản đường đã đi được trong khoản thời gian này.**







* Đường màu đỏ thể hiện con đường mà xe đã đi được trong khoảng thời gian đã chọn, lưu ý cần chọn khoảng thời gian thích hợp để có thể xem được quảng đường mà ta cần dùng tới.
* Mũi tên trên bản đồ thể hiện hướng di chuyển của xe
* Xe chạy nhanh chậm tùy thuộc vào tốc độ của xe và quảng đường đi được, vận tốc của xe càng chậm sẽ làm cho mũi tên di chuyển chậm trong bản đồ.

# ****KẾT LUẬN****

**Đề tài chỉ giới hạn ở mức nghiên cứu công nghệ GPS và phát triển công ngệ TracCar. Từ đó đề xuất ứng dụng của công nghệ GPS và TracCar vào công việc quản lý xe. Tuy nhiên quản lý sẽ có rất nhiều mục để quản lý.**

**Đề tài chỉ mới thực hiện được các công việc sau:**

* **Nguyên cứu công nghệ GPS**
* **Nguyên cứu công nghệ TracCar và phát triển.**
* **Mô hình quản lý xe ứng dụng công nghệ TracCar.**

**Bên cạnh những kết quả đã đạt được, còn có những vấn đề mà thời điểm hiện tại chưa giải quyết được:**

* **Chương trình còn thiếu nhiều tính năng nâng cao để quản lý xe.**
* **Xây dựng một chương trình hoàn chỉnh có giao diện tương tác với người sử dụng**
* **Thời gian thực của xe còn delay cao.**

**TÀI LIỆU THAM KHẢO**

* **Website** <https://www.traccar.org> **ứng dụng TracCar**
* <http://www.objis.com/formationextjs/lib/extjs-4.0.0/docs/index.html> **tài liệu thiết kế giao diện Ext JS**
* Tài liệu javascript <https://www.javascript.com/learn/javascript>
* Tài liệu Java server socket clients <https://docs.oracle.com/javase/tutorial/networking/sockets/clientServer.html>
* Tài liệu google map <https://developers.google.com/maps/>

[4]. Mã nguồn <https://drive.google.com/open?id=1BGwaJjI5pNArlPqYOSBve9hfj5TH4ZOH>