DOKUZ EYLÜL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

DEVELOPMENT OF A GSM AND GPS BASED VEHICLE TRACKING-ROUTING SYSTEM

by

Özkan KAYA

October, 2019

İZMİR

DEVELOPMENT OF A GSM AND GPS BASED VEHICLE TRACKING-ROUTING SYSTEM

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for the Degree of Master of Science
in Geographic Information System, Geographic Information System Program

by
Özkan KAYA

October, 2019
İZMİR

M.Sc THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"DEVELOPMENT OF A GPS AND GPRS BASED VEHICLE TRACKING-ROUTING SYSTEM"** completed by **ÖZKAN KAYA** under supervision of **ASSOC.PROF.DR. OKAN FISTIKOĞLU** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Okan FISTIKOĞLU

Supervisor

Assist. Prof. M. Güven Kocaalp

(Jury Member)

Assoc. Prof. Dr. Ali Gül

(Jury Member)

Prof.Dr. Kadriye ERTEKİN

Director
Graduate School of Natural and Applied Science

ACKNOWLEDGMENTS

First of all, I would like to thank to Assoc. Prof. Dr. Okan Fıstıkoğlu who accepted me as his master student and helped me during this project.

I would like to thank to my wife Elif Kaya and my family who provide full support during the thesis.

Özkan KAYA

DEVELOPMENT OF A GSM AND GPS BASED VEHICLE TRACKING-ROUTING SYSTEM

ABSTRACT

There are a lot of vehicle tracking systems, nowadays. Some of them helps vehicle owners to track their cars on satellite maps that contain street roads, roadway etc. Some of them protect vehicles against thieves, extrem fuel consumption. It is possible to give more examples. Official researches indicate an increasing rate in the essential of GPS tracking systems. But, these systems are not still enough to answer all needs without Geographical Information Systems (GIS).

The scope of this thesis is to develop a GSM-GPS and camera-based vehicle tracking-routing system, with the help of GIS due to above cases. In this regard if to need to summarize the thesis; It was examined mounting a camera to vehicle kit for some security reasons and routing of the vehicle. It was investigated how to use vehicle kit with GIS software together via “Map Server” to find solutions about new problems for car owners and fleet managers. Besides all of them, it was designed a vehicle control web pages substantially.

Consequently, developing a vehicle tracking-routing system can be considered a faster and safer option for minimizing theft crimes, monitoring vehicles and new problems with GIS techniques.

Keywords: Geographic Information Systems, GPS, GSM, camera, relay, vehicle tracking, vehicle routing, relational database, location data, MVC, C#, Google Map, Map Server, Google Apis, RS232, Arduino, Python, C, MySql, Ftp, Http

GSM VE GPS BAZLI ARAÇ TAKİP VE YÖNLENDİRME SİSTEMİNİN GELİŞTİRİLMESİ

ÖZ

Günümüzde birçok araç takip sistemi mevcuttur. Bunlardan bazıları yolları ve sokkaları gösteren uydu haritaları yardımıyla araçların harita üzerinden izlenmesini sağlar. Bazıları araçların çalınmasını ya da daha falza yakıt tüketmesine karşı çözüm olabilir. Araç takip sistemlerinin faydalarını saymakla bitmez. Bu yönde ki resmi araştırmalar araç kitinin gerekliliğini ve kullanım oranının fazlalaştığını göstermektedir. Yinede tüm bu sistemler hâla Coğrafi Bilgi Sistemleri (CBS) olmadan tüm ihtiyaçları karşılayamamaktadır.

Bu tezin kapsamı yukarda bahsi geçen uygulamaları ve yeni ihtiyaçları CBS yardımıyla karşılayabilecek GSM-GPS ve kamera tabanlı geliştirilebilir araç takip ve yönlendirme sistemi üzerinedir. Bu bağlamda çalışmayı özetlersek; bazı güvenlik ve araçların yönlendirilmesi nedeniyle geliştirilebilir araç kitine bir kamera eklenmesi incelenmiştir. Yine araç sahiblerine ve filo yöneticilerine gelecekteki yeni problemlere çözüm bulabilmeleri için “MapServer” aracılığı ile araç kitinin CBS yazılımları ile birlikte nasıl kullanılabileceği araştırılmıştır. Tüm bunların yanında bir araç kontrol internet sayfası gerçek anlamda tasarlanmıştır.

Sonuç olarak, geliştirilen araç izleme-yönlendirme sistemi ile hırsızlık suçlarını en aza indirebilecek, araçaları CBS teknikleri ile gerçek zamanlı izleyebilecek ve yeni problemeleri basit bir şekilde çözebilecek daha hızlı ve güvenli bir araç takip sistemi geliştirilmiştir.

Anahtar kelimeler: Cografi Bilgi Sistemleri, GPS, GSM, kamera, röle, araç takip, araç yönlendirme, ilişkisel veri tabanı, konum verisi, MVC, C#, Google Map, Map Server, Google Apis, RS232, Arduino, Python, C, MySql, Ftp, Http

CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

CHAPTER ONE
INTRODUCTION

1.1 Research Objectives

The main purpose of this thesis investigates relation between a developable vehicle tracking systems and GIS techniques, and presents them with applications.

Many private and governmental companies and institutions need to remotely control their vehicles depending on their purposes. GIS has tried to find solutions for these purposes in this thesis.

GIS has tried finding solutions for companies not demanding to share their vehicle's location information with a second company. Gold or Metallic Mine Companies can be example if need to give an example.

Additionally, transforming information, which is obtained using GIS methods, from geographic coordinate systems to projected coordinate systems and using it with related reports has been examined. Using this technique, an attempt has been made to calculate the real speed of the vehicle.

In the case of polygons that is determined as geographically contains vehicles, it aims to be able to reach vehicle quantities in the polygons along with the vehicle registration process.

It aims to take a photo inside the vehicle by assembling a camera inside the vehicle. It aims to increase seatbelt usage with this technique.

1.2 Existing Vehicle Tracking Systems

Vehicle tracking systems consist of a GPS (Global Positioning System) – GSM (Global System for Mobile Communications) communication kit that is mounted inside the vehicle, a vehicle control web page and a map server. These systems that only monitored vehicles from remote with one way communication in the past allows

the dispatch and control of the vehicle by getting in contac with the vehicle now. Sometimes, vehicle tracking systems that are developed for plenty of purposes using to get under control of fuel consumption against increasing petrol prices (Chadil, Russameesawang & Keeratiwintakorn, 2008), they assistance smart transportation systems sometimes, as well (Mcdonald, 2006). Vehicle tracking systems, which are widely used by cargo companies for tracking cargo to the users, remotely routing of lifeguard vehicles such as fire and ambulance, have become a system that is used in the prevention of theft by attaching to vehicles for security purposes (Maurya, Singh & Jain, 2012). Thanks to developing mobile technologies, providing all these services via smartphones have expanded applicable fields of vehicle tracking-routing systems (Lee, Tewolde & Kwon, 2014).

Today, many private and government organizations feel the need to remotely monitor, control and direct their vehicles, depending on their operational objectives. To this end, they provide for their needs through the purchase of services from many domestic or foreign companies that provide such services. However, because of security, strategic or cost reasons, some companies are trying to solve these needs by using their own technic background, not through service procurement. In addition, these companies aim to create more advanced systems by integrating information systems that are important in their field of activity with vehicle tracking systems (Khoury & Zgheib, 2018). For example, jewelry companies or courier companies that collect cash from multi-branch markets demand to perform the monitoring of their vehicles themselves instead of taking the service and seek ways to integrate additional modules such as shortest route analysis and accounting statistics into the monitoring systems.

Support of a vehicle tracking and routing system suitable for modular expansion with GIS enables system developers to gain GIS's ability to process spatial data into vehicle tracking systems while increasing the variety of application of the system (Davis, 2007). Through GIS vehicle tracking systems, spatial data can be used much more efficiently, making spatial analysis on these data helps in the correct decision-making time (Stillwell & Clarke, 2004).

In the present study, hardware and software-sized design of a vehicle tracking system which can be completely redesigned according to the needs of the user is given. The system designed includes a GSM-GPS communication kit, a user interface that has a web site that allows monitoring positions of vehicle and calculating GIS analysis, a map server that publishes all these services as graphically.

The system that offered calculates and reports real speed association with the location by converting vehicle location information in geographical coordinates to projected coordinates.

Leading vehicle tracking systems in Turkey:

Arvento, Triomobil, Turkcell Vehicle Tracking and Vodafone Vehicle Tracking are leading vehicle tracking systems. All these companies give live vehicle tracking services via web pages. While GPS and speed information are the basic services provided, they are able to route vehicles with temperature, humidity and relay sensors based on their needs.

1.3 Place and Importance of GIS in Vehicle Tracking Systems

Managing vehicles using GIS is rather easy. In addition to all these, it is more important to quickly locate the vehicle's position on the map in two dimensions. Querying and reporting the vehicle using GIS according to time, place and speed means being reliable and faultless. Only fast and reliable querying represents the importance of GIS in vehicle tracking systems.

Moreover, the system is easy with GIS to see and route vehicle together waypoints with maps that have different coordinate systems. It is possible to make some geographic analyses and temporally follow it too. The importance of GIS has been emphasised in this thesis' results.

1.4 The Scope of the Thesis

Today, it is easier to design computer software and electronic devices with the ease of developing technology and information access. In the thesis, a vehicle kit, a webpage and a map server were designed for a vehicle tracking-routing system.

The thesis scopes the monitoring and controlling of vehicles using GIS and data transmitted by the vehicle kit.

Informing users via e-mail against vehicles which are stolen, providing WMS services for security situations, determining vehicle quantities with area-determining methods, taking photos inside a vehicle using cameras and speed-warning systems are the base content of this thesis.

In the thesis, the Toyota Corolla model 2011 car was used for the vehicle-tracking system. The system was tested for ten months, collecting about a hundred thousand records. The system has been developed by interpreting the obtained data.

1.5 Thesis Structure (Organisation of the Study)

The thesis consists of siz chapters, organised as the following.

CHAPTER TWO: This chapter shortly explains vehicle tracking system and its parts.

CHAPTER THREE: This section is about controlling and reporting GPS-based vehicle tracking-routing systems with web pages. It is examined by displaying vehicle-speed information and vehicle-location information on Google Maps. It has tried to take a photo inside a vehicle and calculate vehicle quantities parking in fields, which are drawn via the Internet web map page. Vehicle security was discussed.

CHAPTER FOUR: This section is about integrating GPS-GSM-based vehicle tracking systems to GIS software via WMS service. It has tried to show vehicle information in ArcMap and QGIS programs by establishing Map Server in local hosts

(databases and personal PCs). This section also discusses things to do when need not to share vehicle GPS information with a second company and when need to use it with GIS programs.

CHAPTER FIVE: It has tried to design a vehicle kit able to carry out tasks, which is this thesis' purpose. It designed a vehicle kit that is compatible with working with webpages, map services and databases, after the needs were determined. In short, this vehicle kit consists of a camera, and GPS and GSM-based modules. It has examined the choosing of the modules. Additionally, demo applications and important determinations were carried out.

CHAPTER SIX: It has been argued to minimise problems and needs that are emphasised in the purposes of the thesis. It emphasizes the system's advantages and disadvantages. Additionally, it has stressed on the need to open the system to improvement.

CHAPTER TWO
METHOD

It is essential to work vehicle tracking systems with various devices (mobile phone, computer etc.) or GIS software together. This rule is possible with combining a few technologies for vehicle tracking systems. These are respectively a web site, a WMS (Web Map Server) map server and a vehicle kit that is stably working. In this study, designing of a vehicle tracking system that consists of these three technologies was presented. In looking previous studies (Chadil, Russameesawang & Keeratiwintakorn, 2008; Verma & Bhatia, 2013), we always see to work GPS and GSM based vehicle tracking-routing systems on Goolge basemaps. Google's basemaps are used in these systems. But, the systems don not give grants to users to use own maps as the basemaps. In the present study, not only using Google's basemaps and also users can use own special maps as the basemaps with WMS services. The work is presented in the following headings.

2.1 Web Page Progamming Technologies

A web page that was prepared in scope of the study enables the end users to control and route their own vehicles by using functions of powerful web engines that are being developed. In this regard, monitoring location of the vehicle, taking a photo inside the vehicle, transforming coordinates of the vehicle from geographic coordinate system to projected coordinate system and using them to speed calculations, routing the vehicle by drawing polygons on Google's basemaps, are main duties of the vehicle control web page.

Google Map APIs is an important part of the development of the web page. First of all, Google's basemaps are used as basemaps preferred (Svennerberg, 2010). It had been tried to show location address of the vehicle and traffic jam locations on the map by including Google Traffic Api and Google Address Api tools to the web page. Google presents assistant tools to work compatible with HTML and JavaScript for web page developers (Dincer & Uraz, 2013). These tools perform to easily draw a point, a line or a polygon on the web map interface with JavaScript. So, bus stops, traffic signs

or places that is crucial for users are marked as points or polygons and used them for routing of vehicle.

There are several web page development software in our day. Some of these software are paid or free, some have advanced design and encoding interfaces. The presence of strong resources in relation to the use of these software has been one of the reasons for the choice of these software. MVC (Model View Controller) ("ASP.NET Core", n.d.) and RWD (Responsible Web Design) (Aryal, 2014) technologies are commonly preferred (Table 2.1). The superior feature of MVC technology is that dynamic data can be execute with HTML and JavaScript codes (Frain, 2015; Zakas, 2009). The word Model can be shown as a database for programming. The Model covers C# classes that correspond to tables owned by the database. The word View covers design pages that include HTML and JavaScript code. The Controller can be thought of as a engine that allows the exchange of information between the View and the Model. A web project designed with RWD technology make pages more compatible with phones, computers and tablets.

Table 2.1 The comparison of popular web design and encoding technologies

Tecknology	**Model View Controller**	**Price**	**Popular & Documantation**	**Hosting in Turkey**	**Interface Tool**
ASP.Net MVC	Yes	Not Free	Yes	Yes	Advanced
PHP Laravel	Yes	Free	Yes	Yes	Intermediate
Python Django	Yes	Free	Yes	No	Intermediate

Because of the powerful source and coding in this study, ASP.NET MVC software was preferred (Anderson, 2014). On the other hand, ASP.NET is also preferred due safe coding (Jovičić & Simić, 2006). Today, while developing a web page, it is common for the pages to work in harmony with the phone, tablet and computer (Figure

2.1). ASP.NET MVC is powered by HTML5 technology as well (Torre & Carmona, 2013).

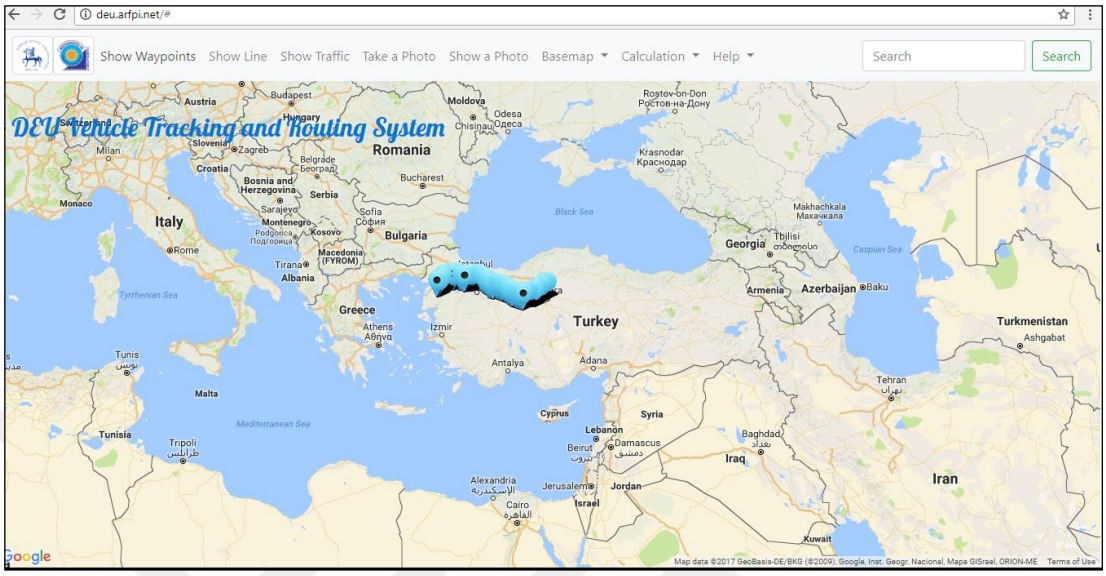

Figure 2.1 Vehicletracking-routing system and ASP.NET MVC (Google, n.d.)

2.2 Web Map Service (WMS) Technology

WMS is a server technology service to service raster maps and vector maps which they are prepared for digital as HTTP formats to GIS software and web browsers with certain standards (Open Geospatial Consortium [OGC], n.d.).

There needs a map server to publish WMS. After this server implements all GIS processes, it publishes maps that they are suitable with WMS standards as image formats on the internet. Table 2.2 summarizes popular map servers.

Table 2.2 Popular map server software

Map Server Technology	**Price**
NETGIS.SERVER.ENT	yes
MapInfo Server	yes
ArcGIS Server	yes
MapServer	no
GeoServer	no

MapServer Software, is a map server, was preferred because of its cost free and common in many communication workings. MapServer is an open source platform to publish GIS files with OGS (Open Geospatial Consortium) rules, was developed by Minnesota University in Mid 1990's ("The MapServer Team", n.d.) (Figure 2.2). Some of its importance features include:

- Support database formats
- Windows, Linux, Mac OS X, etc.
- NET, PHP, Python, Java etc.
- Projection systems
- JPEG, BMP etc. image formats

Figure 2.3 presents informations of vehicle location on the map by adding a HTTP WMS address whose MapServer serviced to GIS software. Mapserver is ready to be published by being installed it on the users' servers or personal computers quickly, and the map can be easily prepared with text editors. Free GIS software such as QGIS, MapWindowGIS etc. have MapServer plugin to prepare WMS maps.

Figure 2.2 MapServer engine official web site

Figure 2.3 ArcMap WMS application with MapServer (Google, n.d.)

2.3 Vehicle Kit Technology

Vehicle kit is heart of the matter of the system.

Vehicle kit, is an electronic module, collects the informations of vehicle location, and makes routing of the vehicle by mounting it to ECU (Engine Control Unit) of the vehicle (Kodavati, 2011). This module consists of two main modules that are GPS and GSM (Verma & Bhatia, 2013). Extra modules such as sensor, camera etc. can develop it more. There are several GSM-GPS modules in vehicle kit technology (Figure 2.4). On the other hand, GSM and GPS correspond with each other over a microcontroller that executes “AT” commands for GSM line (Turkcell Data Card) (Figure 2.5). There are compact and ready vehicle kits instead of all these modules in electronic markets, too.

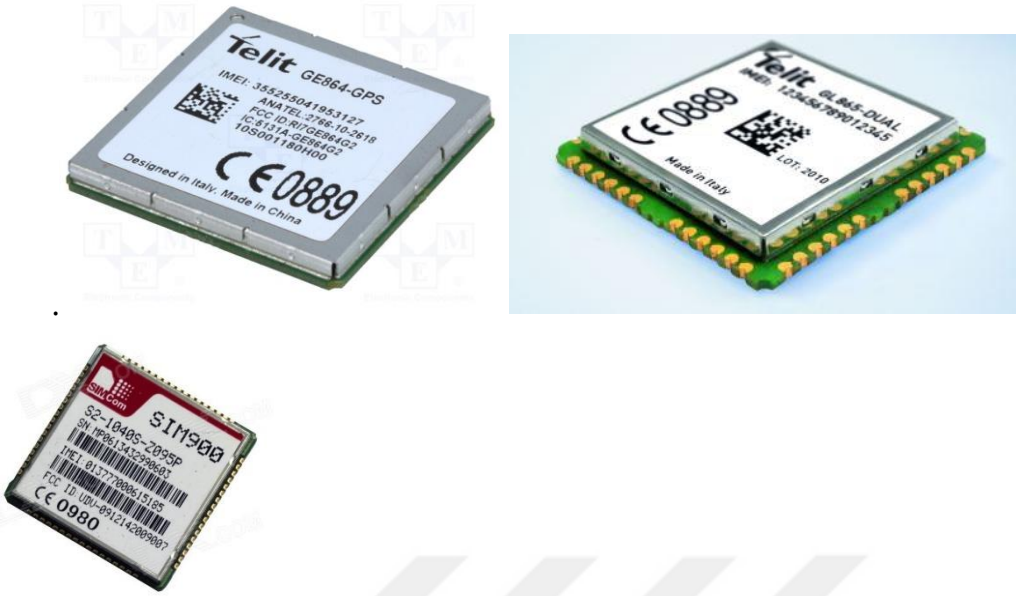

Figure 2.4 Various GMS and GPS brands in the market (Personal archive, 2018)

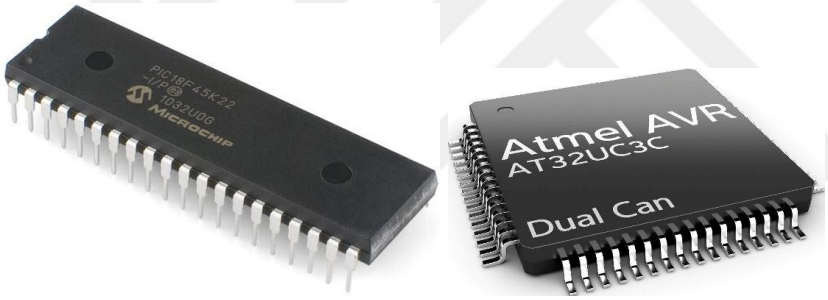

Figure 2.5 Microcontroller samples (Personal archive, 2018)

In this thesis, vehicle kit cariers a camera. Vehicle kit can be different according to aim and needs.

Vehicle kit that consists of electronic modules is expensive. Instead of this, cheaper available vehicle kit that procured is more pratice. But, it is necessary to know vehicle kit that bought. Since, it will be encoded according to aims. Resultly, HE910T-3G GSM-GPS module is preferred in thesis (Figure 2.6). Figure 2.7 explains relationship between vehicle kit and system server as schematic.

Figure 2.6 HE910T-G GSM/GPS module (Personal archive, 2018)

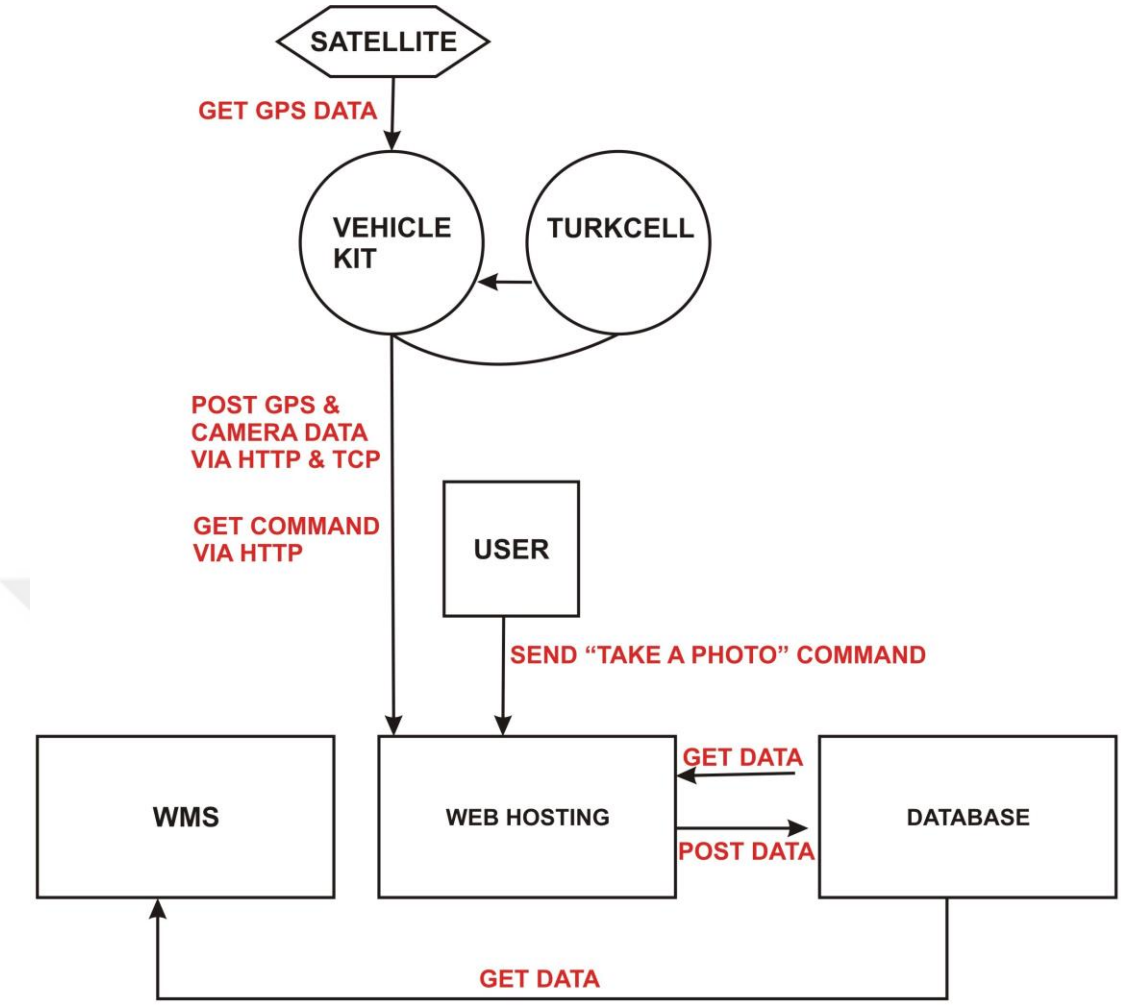

Figure 2.7 The working schema of the vehicle kit with WMS and web hosting

CHAPTER THREE
DEVELOPMENT OF WEB PAGE FOR VEHICLE TRACKING-ROUTING SYSTEM

3.1 The Building of the Web Page

It is very important to quickly and practically develop vehicle tracking-routing systems. It is critical to catch an error and take precaution in the live systems. Webpage design tools should be chosen that will be able to both minimise mistakes and prevent the disruption of the system. In the section one, ASP.NET MVC was preferred (Figure 3.1). Visual Studio 2017 was used for ASP.NET editor in the study.

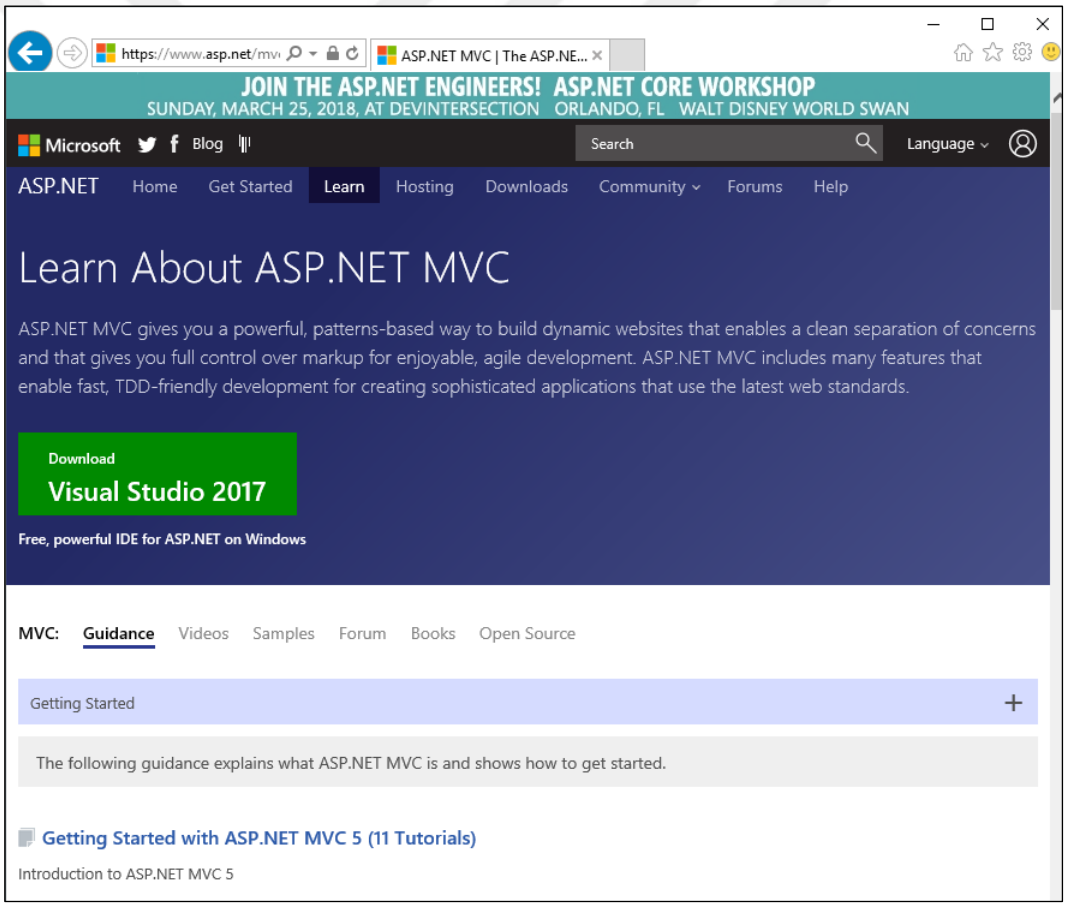

Figure 3.1 ASP.NET MVC official web site

Steps in the below explain basic vehicle tracking web page design with version ASP.NET MVC 5 encoding styles in Visual Studio.

- Creating Models, Views and Controllers folders in the current vehicle project in Visual Studio (3.2).

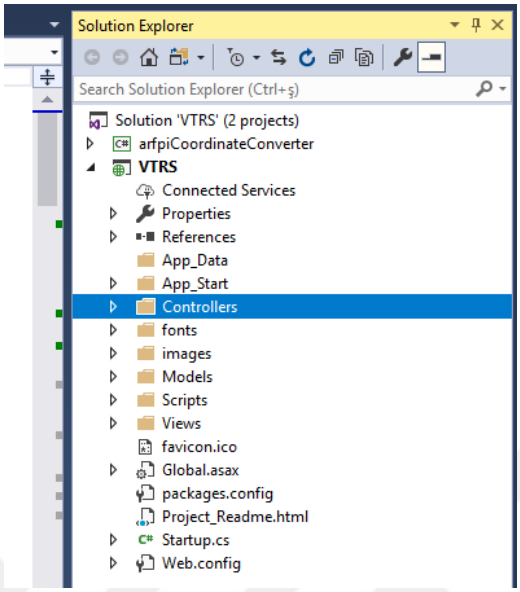

Figure 3.2 Creating Views, Models and Controllers folders in Visual Studio

- MapController in Controllers folder consists of a lot of actions and a few functions and methods (Figure 3.3). More detail about MapController is under appendix sections.

```
public MapController()...
0 references
public ActionResult Password(string Password)...
0 references
public ActionResult Index()...
0 references
public ActionResult SavePolygon(string polygonn,string namee)...
0 references
public ActionResult SaveGPS(string photo,string psw,string lat, string longg,string speed)...
2 references
public string QuantityOfRecord()...
0 references
public ActionResult getGPS(string plateid)...
0 references
public ActionResult getline(string plateid)...
0 references
public ActionResult updateQuantityOfRecord(string quantity)...
0 references
public ActionResult updatePoly(int row,string email)...
0 references
public ActionResult getPolygon()...
0 references
public ActionResult GetPhoto()...
0 references
public ActionResult TakePhoto()...
1 reference
public byte[] hexToByteArray(string hexString)...
1 reference
private byte[] ReadImage()...
0 references
public ActionResult CalculateED50()...
0 references
public ActionResult CalculateSpeed(int id1,int id2)...
0 references
public void sendEmail(string email)...
0 references
public ActionResult CalcSpd()...
```

Figure 3.3 MapController in Visual Studio

- Views folder consists of Map and Shared folders (Figure 3.4). Figure 3.5 indicates 'Index.cshtml' web page. More detail about Views and its codes is under appendix sections.

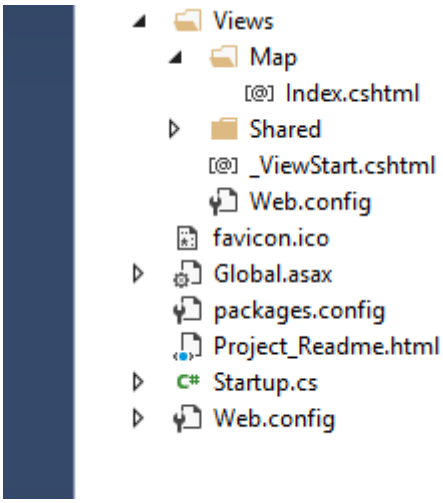

Figure 3.4 Views in Visual Studio

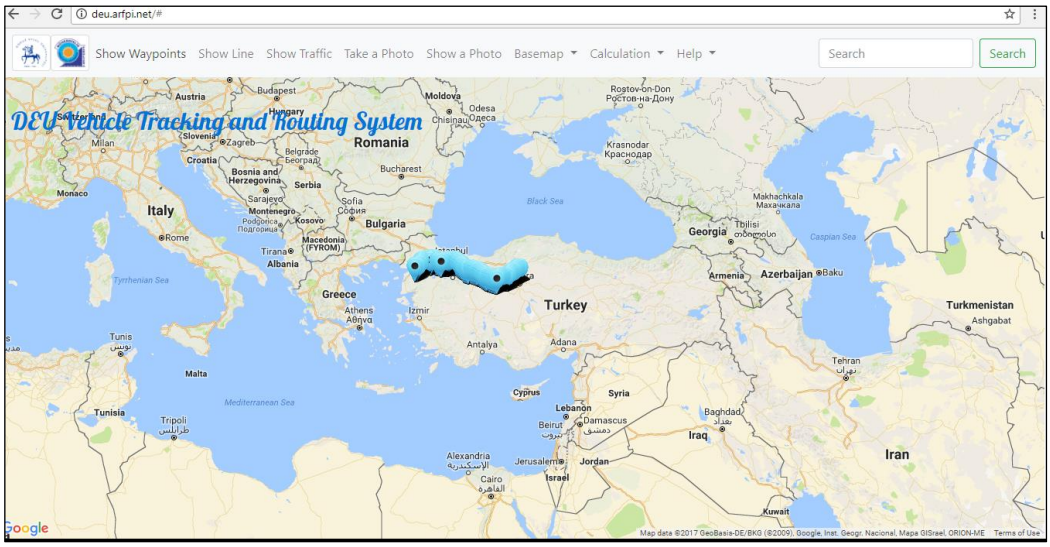

Figure 3.5 Vehicle tracking system (Google, n.d.)

- Models folder includes various classes association with MySQL tables (Figure 3.6).

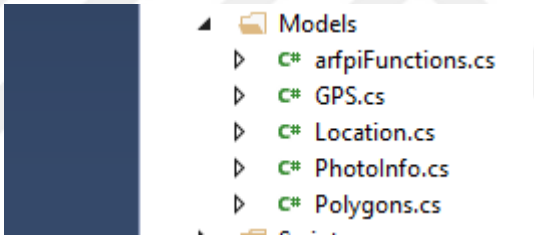

Figure 3.6 Models in Visual Studio

3.2 Using Google Maps in Vehicle Tracking System

Google provides helping tools with HTML and JavaScript for webpage developers. One of them is the Google Maps tool. Google Map HTML codes can be easily added to the webpage as follow (Figure 3.7) (https://developers.google.com/maps/).

Google official web site includes a lot of codes and scenarios for map applications.

```
<!DOCTYPE                                                    html>
<html>
                                                              <head>
                                  <title>Simple                Map</title>
          <meta     name="viewport"     content="initial-scale=1.0">
                                    <meta                 charset="utf-8">
                                                              <style>
```

```
      /* Always set the map height explicitly to define the size
of                                the                                div
               *    element    that    contains    the    map.    */
                                                  #map                {
                                                height:           100%;
                                                                      }
       /* Optional: Makes the sample page fill the window. */
                                          html,          body          {
                                                height:           100%;
                                                  margin:            0;
                                                 padding:            0;
                                                                      }
                                                                </style>
                                                                 </head>
                                                                  <body>
                                      <div                id="map"></div>
                                                                <script>
                                                   var                map;
                                    function          initMap()         {
        map = new google.maps.Map(document.getElementById('map'),
{
                      center:    {lat:    -34.397,    lng:    150.644},
                                                        zoom:          8
                                                                    });
                                                                      }
                                                               </script>
                                                                <script
src="https://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&ca
llback=initMap"
                                      async               defer></script>
                                                                </body>
</html>
```

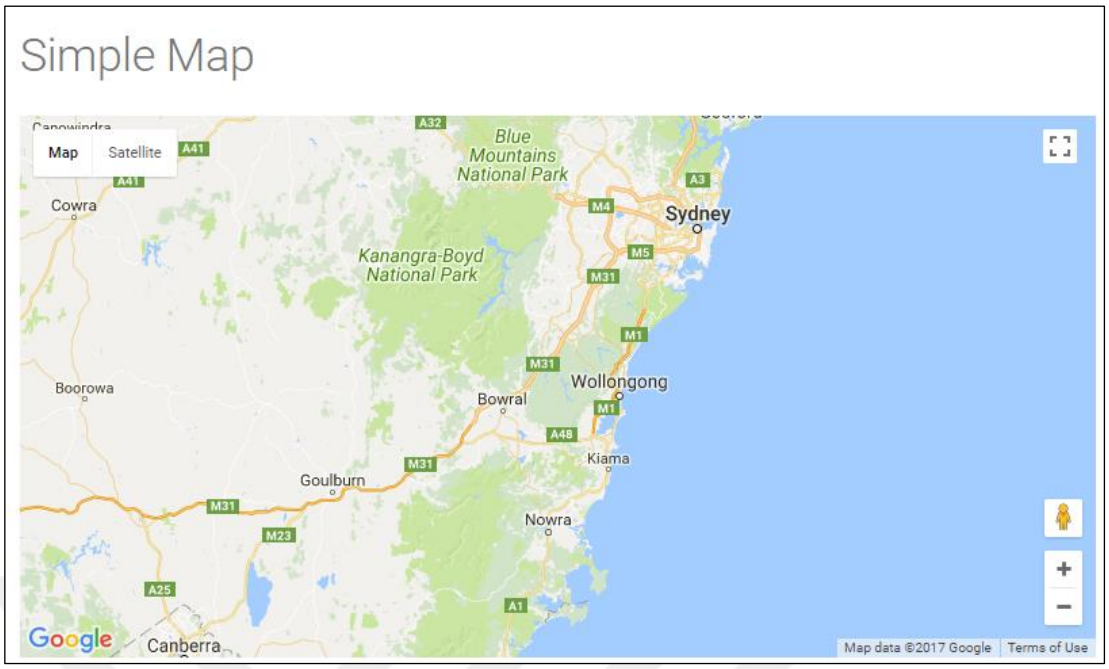

Figure 3.7 Google Map APIs (Google, n.d.)

3.3 Basic Database for Vehicle Tracking-Routing System

In the vehicle tracking-routing system database, a minimum of two tables are needed to keep vehicle location information and vehicle properties Both tables are linked to each other respective plate licences that are unique. In the thesis, MySQL is preferred as database for its power and flexibility (Norman, 2006).

One of the tables is the ‘deucarlog’ table that displays the location and the speed records of vehicles, every five seconds (Table 3.1). The second table is the ‘deucars’ table where either the vehicle features or commands, coming from clients, are kept (Table 3.2).

Table 3.1 ‘deucarlog’ table which stores the geographic data of the car positions

Name	Type	NULL	Default
Id	Int(11)	No	<auto_increment>
Latitude	Double	Yes	
Longitude	Double	Yes	
Plate	Varchar(40)	Yes	
Speed	Double	Yes	
Date	Varchar(20)	Yes	
EastingED50	Double	Yes	
NorthingED50	Double	Yes	
RealSpeed	Double	Yes	
UTM	Int(11)	Yes	

Id	Latitude	Longitude	Plate	Speed	Date	EastingED50	NorthingED50	RealSpeed	UTM
64437	26.89397	39.98806		0.03	20171122 09:12:49	490947.3247073	4426513.240530	<NULL>	35
64436	26.89397	39.98804		0.01	20171122 09:12:40	490947.3220674	4426511.020666	<NULL>	35
64435	26.89397	39.98804		0	20171122 09:12:31	490947.3220674	4426511.020666	<NULL>	35
64434	26.89397	39.98804		0	20171122 09:12:23	490947.3220674	4426511.020666	<NULL>	35
64433	26.89397	39.98804		0.02	20171122 09:12:14	490947.3220674	4426511.020666	<NULL>	35
64432	26.89397	39.98804		0	20171122 09:12:05	490947.3220674	4426511.020666	<NULL>	35
64431	26.89397	39.98804		0.01	20171122 09:11:56	490947.3220674	4426511.020666	<NULL>	35
64430	26.89397	39.98804		0.01	20171122 09:11:48	490947.3220674	4426511.020666	<NULL>	35
64429	26.89395	39.98803		0.03	20171122 09:11:39	490945.6131777	4426509.912766	<NULL>	35
64428	26.89395	39.98803		0.03	20171122 09:11:30	490945.6131777	4426509.912766	<NULL>	35
64427	26.89395	39.98803		0.01	20171122 09:11:22	490945.6131777	4426509.912766	<NULL>	35
64426	26.89395	39.98803		0.02	20171122 09:11:13	490945.6131777	4426509.912766	<NULL>	35
64425	26.89395	39.98803		0	20171122 09:11:04	490945.6131777	4426509.912766	<NULL>	35
64424	26.89395	39.98803		0	20171122 09:10:56	490945.6131777	4426509.912766	<NULL>	35
64423	26.89395	39.98803		0.01	20171122 09:10:47	490945.6131777	4426509.912766	<NULL>	35

Table 3.2 ‘deucars’ table which stores properties of car

Name	Type	NULL	Default
Id	Int(11)	No	<auto_increment>
Photo	Varchar(50)	Yes	
Plate	Varchar(50)	Yes	
Owner	Varchar(50)	Yes	

Id	Photo	Plate	Owner
1	clear***	<NULL>	Özkan Kaya

GPS-coordinated data is received in the WGS84 format from satellites. Coordinates in Turkey should be converted to the UTM ED50 coordinate system to get a correct and reliable projection. There are four different UTM zones in Turkey: UTM 35, UTM 36, UTM 37 and UTM 38, respectively. The ‘deuutm’ table, consisting of UTM zone polygons, has been created to transform process (Table 3.3). Using the table, the coordinates, which are received as degrees can be used to know which UTM zone is at the point of intersection. Then, the transformation parameters of coordinates that pair with UTM zones are found on ‘libsrids’ table (Table 3.4).

Table 3.3 ‘deuutm’ table which contains UTM zone polygons

Name	Type	NULL	Default
UTM	Int(11)	Yes	
GIS	Geometry	Yes	
Id	Int(11)	No	<auto_increment>

UTM	GIS	id
25	<GEO>	267
26	<GEO>	268
27	<GEO>	269
28	<GEO>	270
29	<GEO>	271
30	<GEO>	272
31	<GEO>	273
32	<GEO>	274
33	<GEO>	275
34	<GEO>	276
35	<GEO>	277
36	<GEO>	278
37	<GEO>	279
38	<GEO>	280
39	<GEO>	281
40	<GEO>	282

Table 3.4 ‘libsrids’ table which contains coordinate systems

Name	Type	NULL	Default
Id	Int(11)	No	<auto_increment>
SRID	Varchar(10)	Yes	
CRDSYS	Longtext	Yes	

id	SRID	CRDSYS
8014	2000	<MEMO>
8015	2001	<MEMO>
8016	2002	<MEMO>
8017	2003	<MEMO>
8018	2004	<MEMO>
8019	2005	<MEMO>
8020	2006	<MEMO>
8021	2007	<MEMO>
8022	2008	<MEMO>
8023	2009	<MEMO>
8024	2010	<MEMO>
8025	2011	<MEMO>
8026	2012	<MEMO>
8027	2013	<MEMO>
8028	2014	<MEMO>
8029	2015	<MEMO>
8030	2016	<MEMO>

The fifth and last table is the ‘deupolygons’ table. When vehicle locations intersect with polygons in the table, it can send warning e-mails to related users through the system or can report vehicle quantities in areas as time-dependent.

Table 3.5 ‘deupolygons’ table which stores user polygons

Name	**Type**	**NULL**	**Default**
Id	Int(11)	No	<auto_increment>
PolygonName	Varchar(50)	Yes	
GIS	Geometry	Yes	

3.4 The Using of the Web Page

The demo web page can be accessed from http://deu.arfpi.net. The menu bar developed for the web page is given in Figure 3.8.

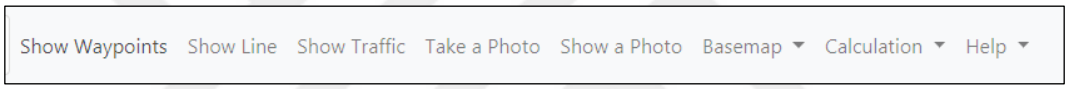

Figure 3.8 Webpage menus

3.4.1 Menus

Show Waypoints: This displays the vehicles’ location information as bubble icons on the map (Figure 3.9). It displays the last thousand records by deleting previous location points whenever it is clicked.

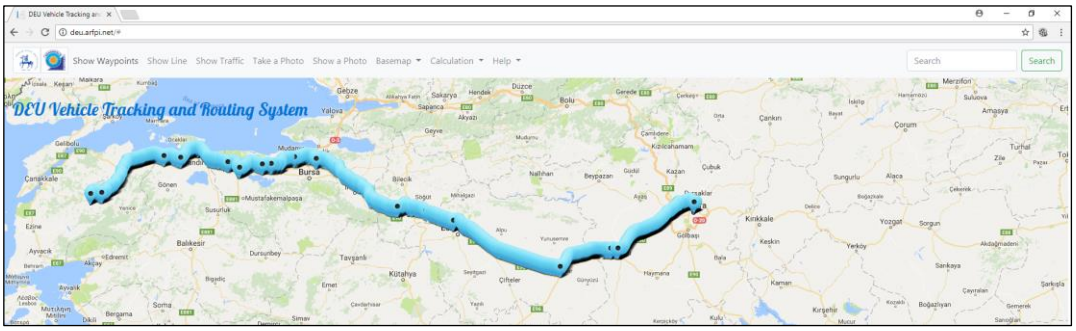

Figure 3.9 Displaying the waypoints of vehicles on the map as bubble icons (Google, n.d.)

Show Line: It displays the direction of the waypoints as a red line and red arrows (Figure 3.10).

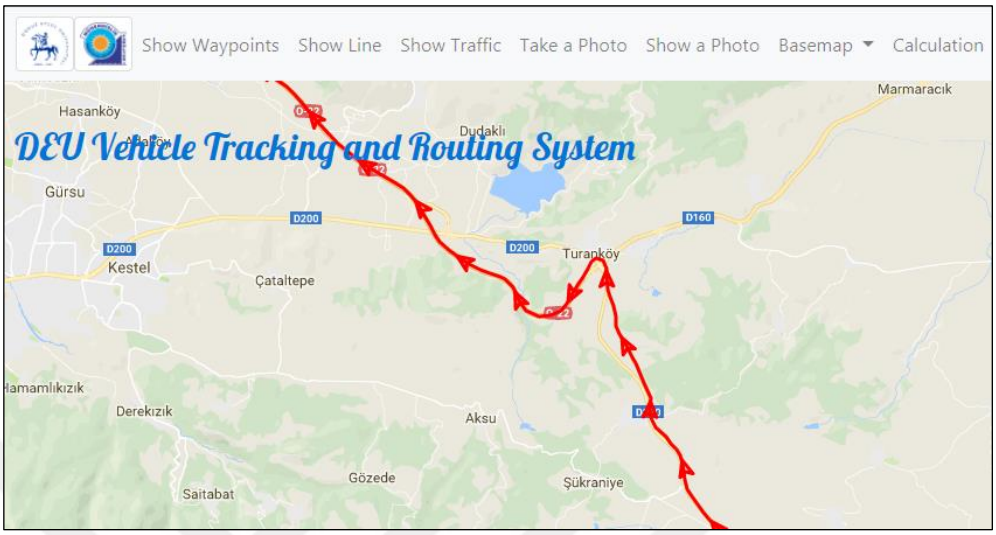

Figure 3.10 The using of the 'Show Line' menu (Google, n.d.)

Show Traffic: It allows the user to watch live traffic jams where vehicles are tracked (Figure 3.11).

Figure 3.11 The using of the 'Show Traffic' menu (Google, n.d.)

Basemap: It provides the option of choosing Road Map, Satellite Map or Hybrid Map as the basemap (Figure 3.12 and Figure 3.13).

Figure 3.12 Satellite basemap screenshot (Google, n.d.)

Figure 3.13 Hibrit basemap screenshot (Google, n.d.)

Calculation→Calculate Geographic WGS84 to ED50: This transforms the WGS84 coordinate system to a ED50 coordinate system (Figure 3.14).

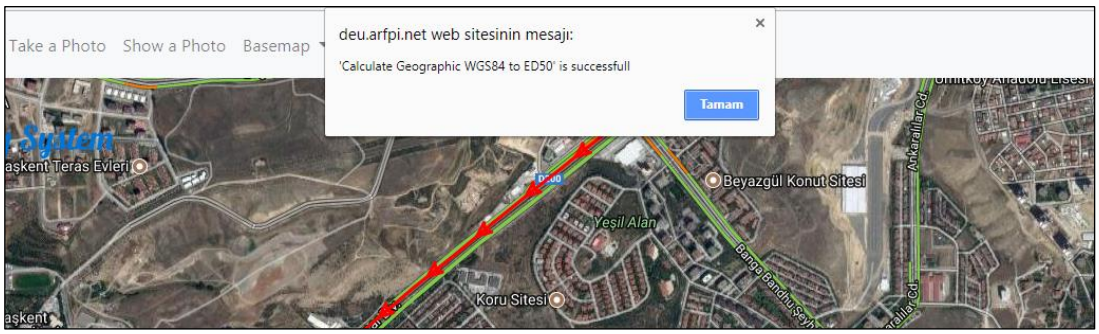

Figure 3.14 The Using of the 'Calculate Geographic WS84 to ED50' menu (Google, n.d.)

Calculation→Draw a Polygon: When this menu is clicked, it starts to draw a polygon on the map. The mouse icon will be a cross icon (Figure 3.15).

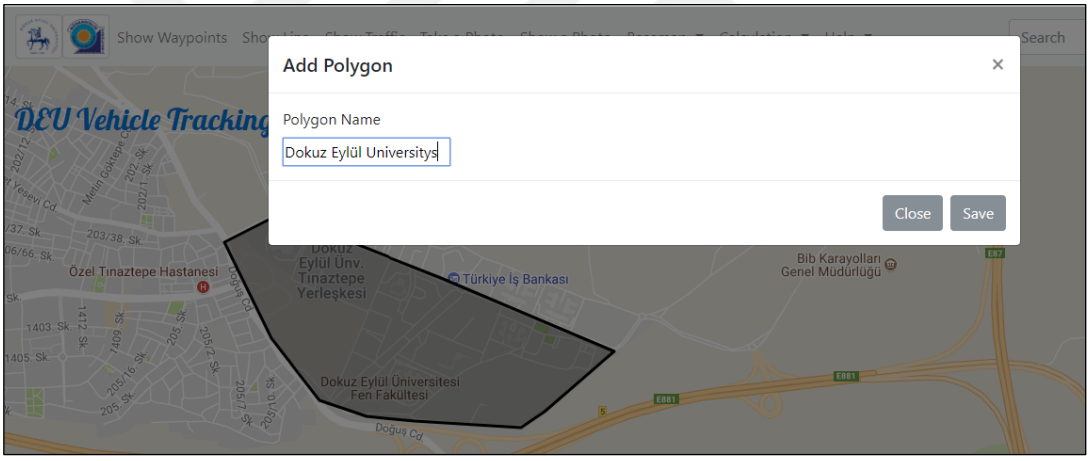

Figure 3.15 The using of the 'Draw a Polygon' menu (Google, n.d.)

Calculation→Show Polygons: It displays polygons that are drawn by the 'Draw a Polygon' menu (Figure 3.16).

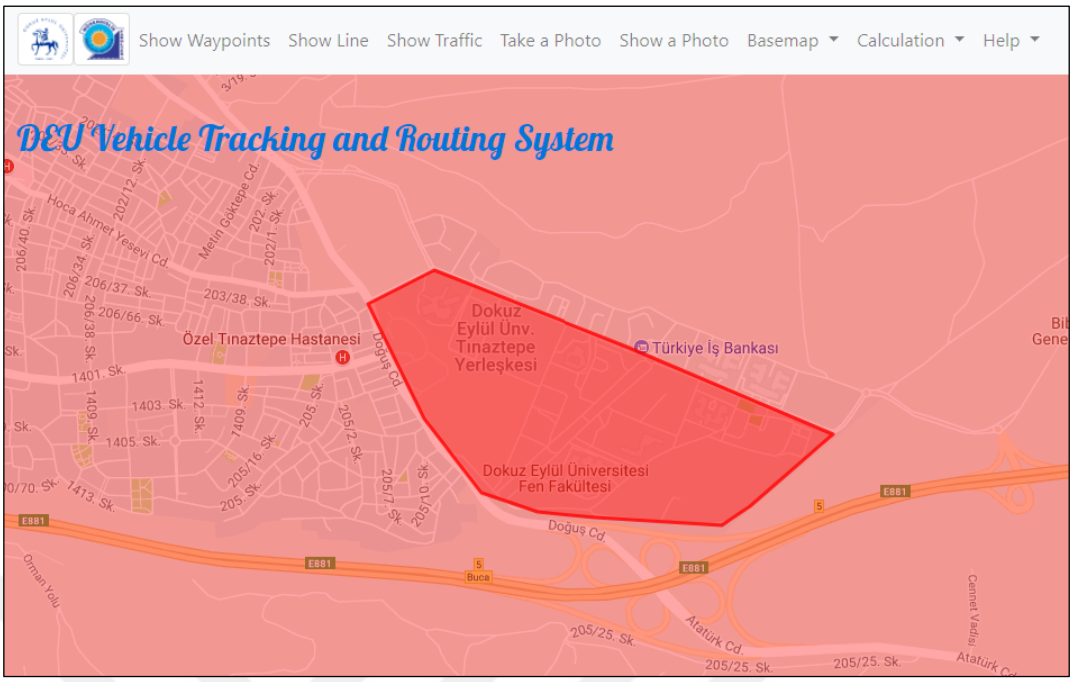

Figure 3.16 The using of the 'Show Polygons' menu (Google, n.d.)

Clicking Menu: It defines an e-mail address to a clicked polygon (Figure 3.17). When any vehicles defined in the system pass on related polygons, the system sends e-mail, previously defined in the system, to the related user. This GIS method can be useful for service vehicles.

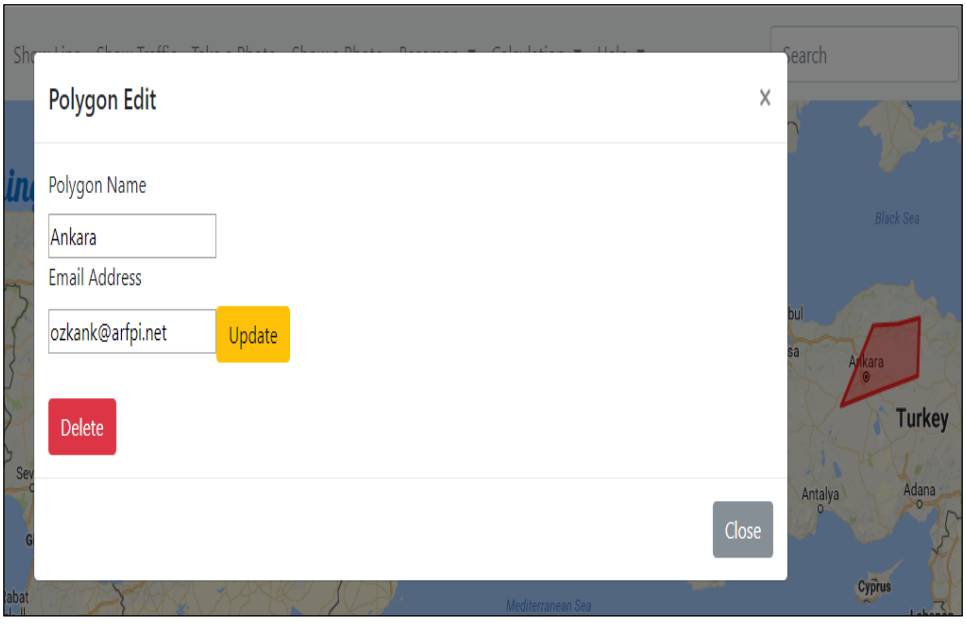

Figure 3.17 Defining an e-mail address to the polygon (Google, n.d.)

3.4.2 Viewing of Vehicle Information

It is important to show GPS speeds, coordinates and moving times of vehicles on the map. Therefore,

1- The waypoints of vehicles are displayed as bubble icons by clicking the ‘Show Waypoints’ menu.
2- When the user needs to see the vehicle speed, the bubble icon on the map is clicked (Figure 3.18).

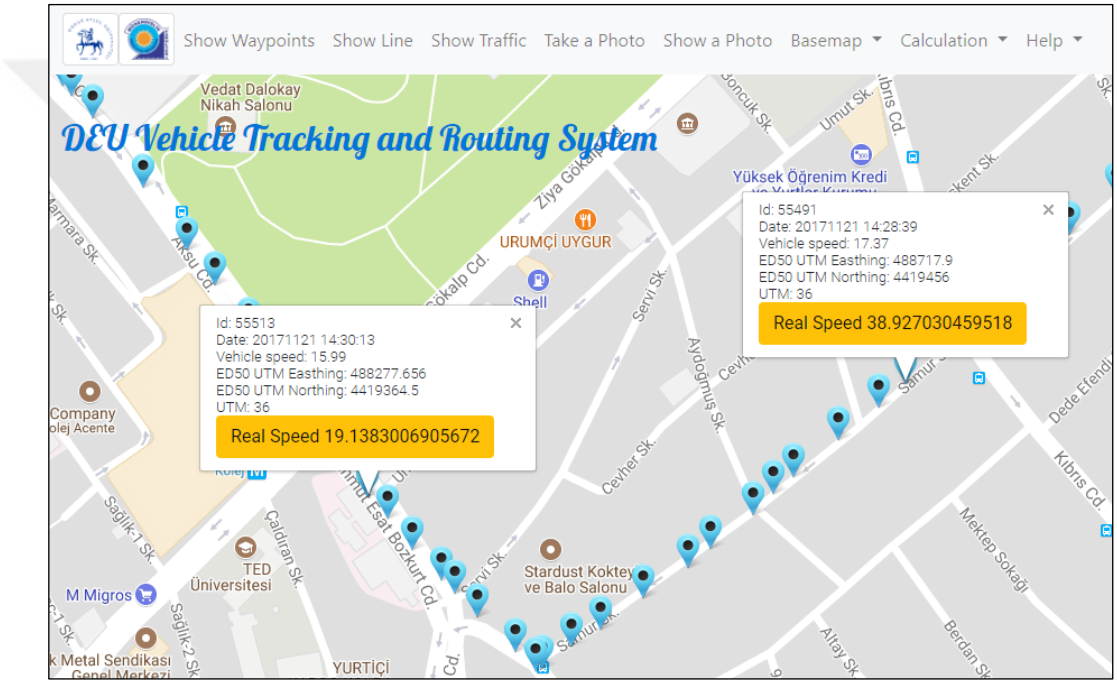

Figure 3.18 Showing vehicle information on map (Google, n.d.)

3.4.3 Taking a Photo Inside the Vehicle

A photo taken inside a vehicle can be used for various purposes, Including for the use of security or monitoring. A photo can be taken inside a vehicle by including a camera to the developed vehicle kit.

1- Open vehicle tracking-routing webpage.
2- Click the ‘Take a Photo’ menu to send the ‘taking a photo’ command to the vehicle kit (Figure 3.19).

3- Click the ‘Show a Photo’ menu to display the last photo taken inside the vehicle on the webpage (Figure 3.20).

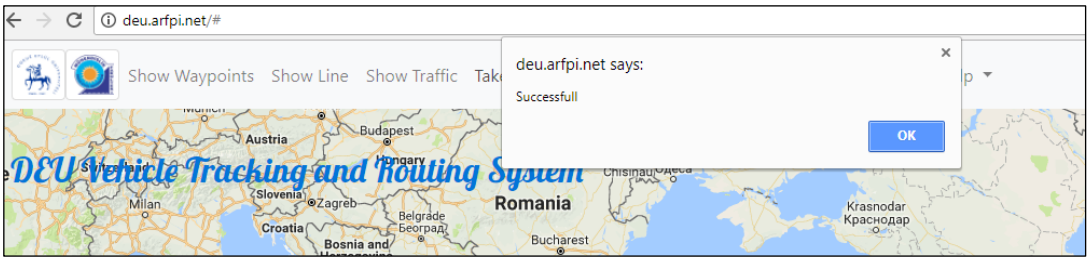

Figure 3.19 The using of the ‘Take a Photo’ menu (Google, n.d.)

Figure 3.20 The using of the ‘Take a Photo’ menu (Personal archive, 2018)

CHAPTER FOUR
TO DISPLAY VEHICLE IN GIS SOFTWARE USING WMS

4.1 Using WMS in Vehicle Tracking-Routing System

The best way of using WMS and it importance in a vehicle tracking-routing system is emphasised with a demo application.

1- Open the vehicle tracking-routing system webpage.
2- Click the 'Help→How to Connect WMS Server' menu. The link describing what needs to be added to the GIS software comes as a warning window on the screen. (Figure 4.1).

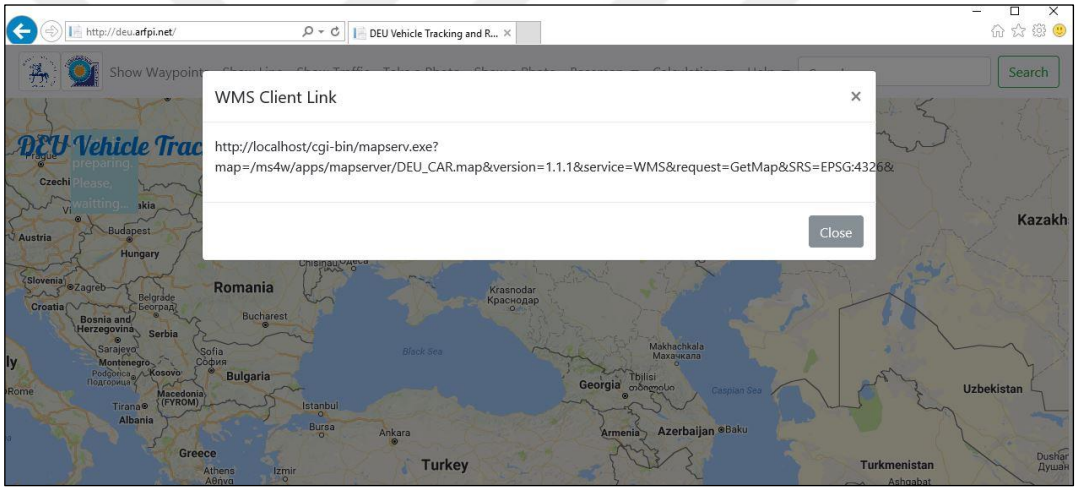

Figure 4.1 WMS client link for GIS software (Google, n.d.)

3- Copy the link from the dialog window.
4- Open the ArcMap software.
5- Click the 'Add WMS Server' link from the Catalog (Figure 4.3).

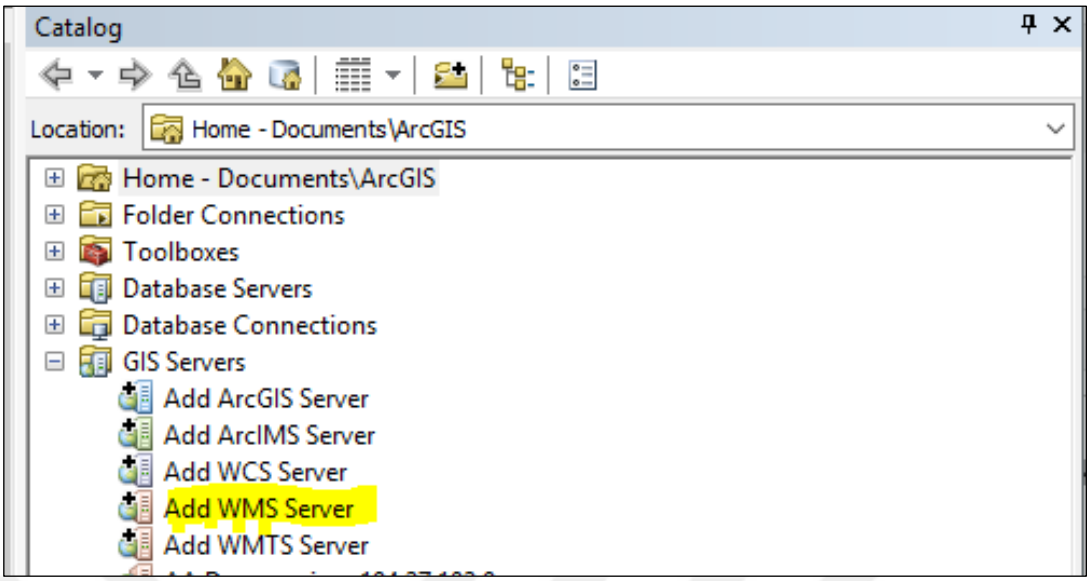

Figure 4.2 'Add WMS Server' link in ArcMap

6- Paste the link into the URL textbox and click the 'GetLayers' button.

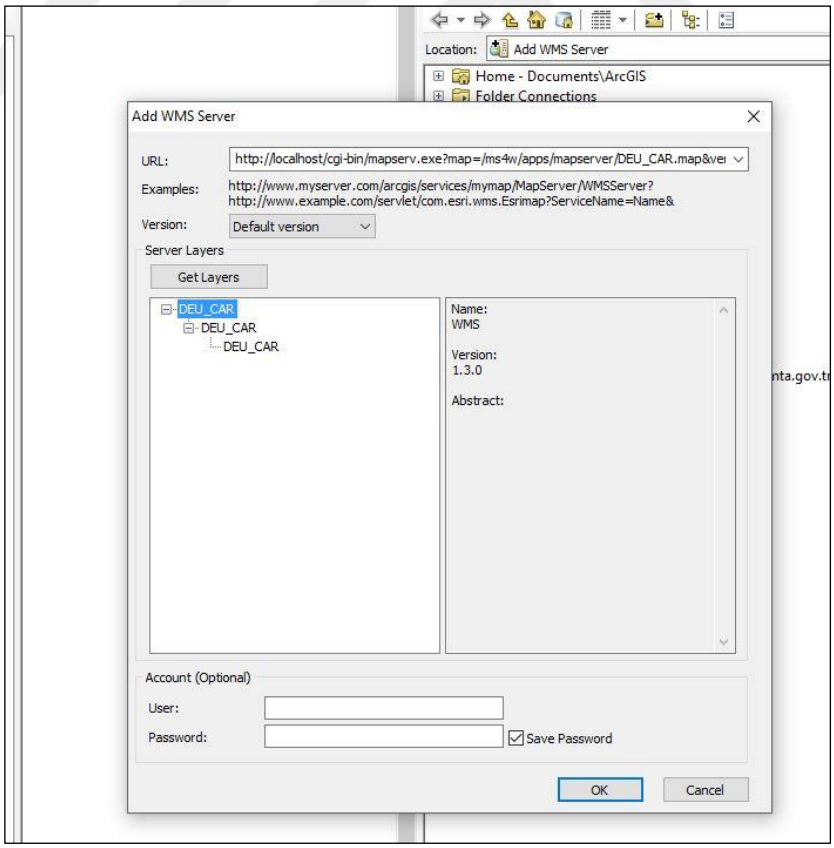

Figure 4.3 Add WMS server dialog window

7- ArcGIS creates a shortcut link for WMS as shown in the Figure 4.4.

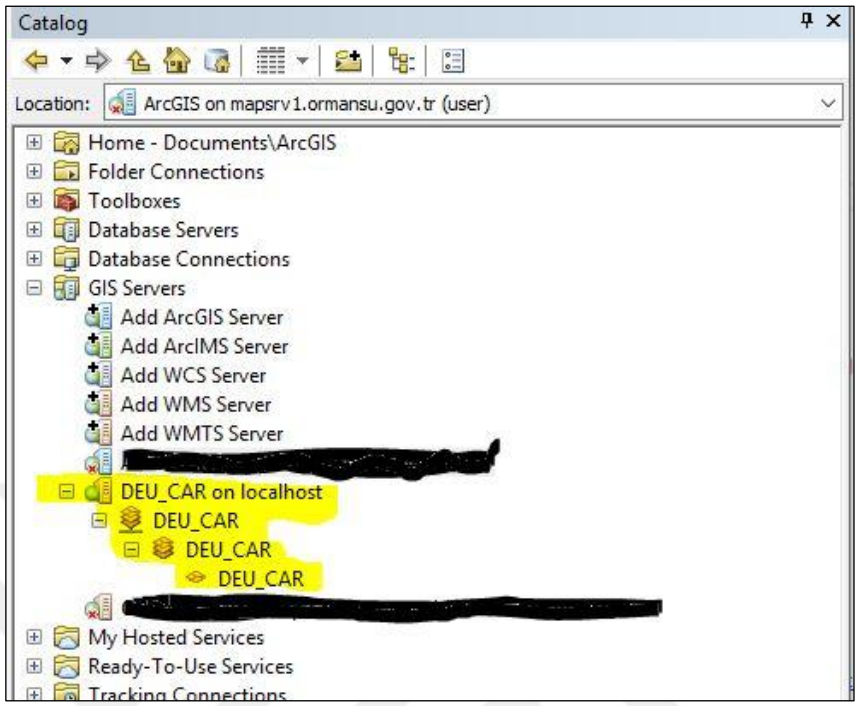

Figure 4.4 WMS shortcut for DEU vehicle tracking-routing system

8- Select the shortcut link. Drag and drop the link onto the map screen.

9- Add satellite basemap to the map by clicking the small arrow icon near ‘Add layer’ icon (Figure 4.5).

Figure 4.5 ArcGIS WMS application with MapServer

CHAPTER FIVE
DEVELOPMENT OF VEHICLE KIT

5.1 Electronic Modules Used to Build the Vehicle Kit

The electronic modules used to build the vehicle kit are as follows:

1- HE910T-3G GSM and GPS Module
2- VC706 RS232 Camera Module
3- Arduino Uno Module
4- RS232 Converter Module

They are examined as the following respective sub-headings:

5.1.1 HE910T-3G GSM and GPS Module

The GSM/GPS module receives its location coordinates and speed information from the satellite using the Telit mark GPS device, and sends all the GPS data to the MySQL database over the HTTP protocol using the Telit mark GSM device. The Module has 2-megabyte memory and 2.7 version Python programming language ("HE910 Family Product Description", n.d.), which provides fast and powerful programming. Its 3G feature, its connection almost always keeps it live (Figure 5.1). Table 5.1 indicates HE910T technic properties.

Table 5.1 HE910T-3G GSM and GPS module ("HE910 Family Product Description", n.d.)

Dimension	:	85 x 70 x 33 mm
Terminal Type	:	UMTS (3G) Terminal
Band	:	Penta-band EGSM 850/900/1800/1900/2100 MHz
Supply Voltage Range	:	9V-32V
Operation Temperature	:	-40 ℃ to +85 ℃
Weight	:	140gr
Embedded Programming	:	Yes (with Python 2.7)
Control via AT commands	:	GSM 27.005 ,27.007 and Telit custom AT Commands
Interfaces	:	SIM, RS232, 4 I/O Ports, Antenna, 1 A/D, Analog Audio Option, USB
Other Features	:	Run AT Remotely, Embedded FTP & SMTP Client, Embedded TCP/IP Stack
GPS Option	:	Yes
WatchDog Option	:	No

Figure 5.1 HE910T-3G GSM/GPS module (Personal archive, 2018)

5.1.2 VC706 RS232 Camera Module

The VC706 camera module's properties ("LinkSprite", n.d.):

- Video output: CVBS 30fps
- Working temperature: -20°C - +60°C
- Humidity: 90% non-condensing
- Working voltage: DC4.8V - DC6.5V

The VC706 camera carries a Max232 chip that supplies communication with the HE910-3G GSM/GPS module (Figure 5.2).

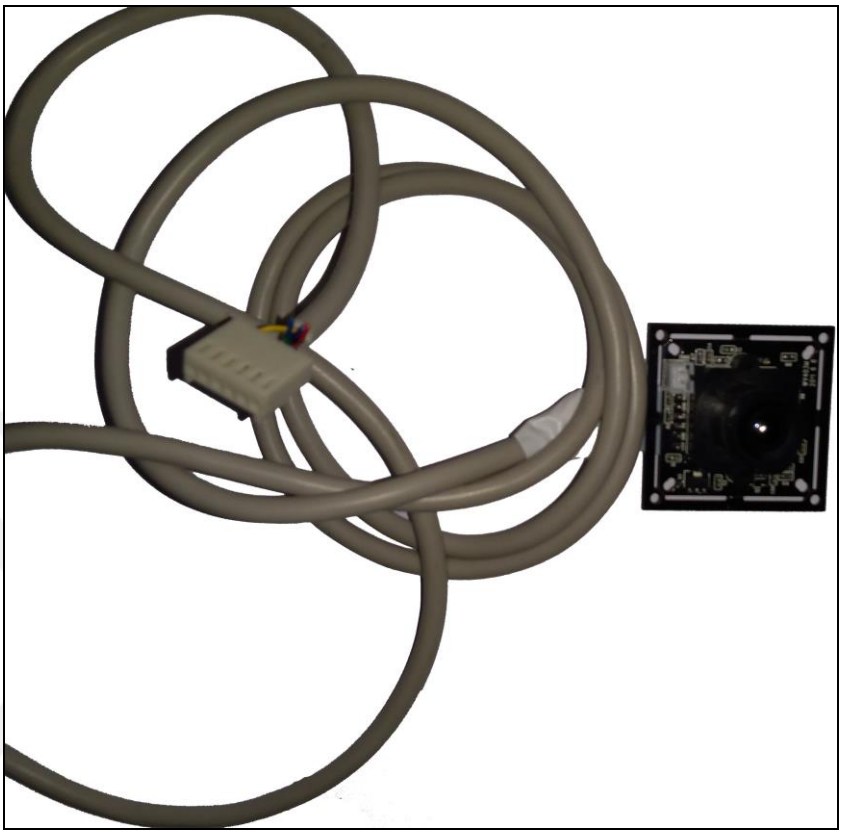

Figure 5.2 VC706 RS232 camera module (Personal archive, 2018)

5.1.3 Arduino Uno Module

Arduino is an open-source platform for electronic developers with developer interface in the digital world.

In the thesis, Arduino Uno, which is encoded by the C++ program language, was used to read data from the VC706 camera module and to send these records to the RS232 converter module. However, before Arduino sends the camera data, it sends a few commands to the VC706 to recieve hexadecimal data from it. Then it orders them between FFD8 and FFD9. Finally, it sends all the records to the VC706 (Figure 5.3).

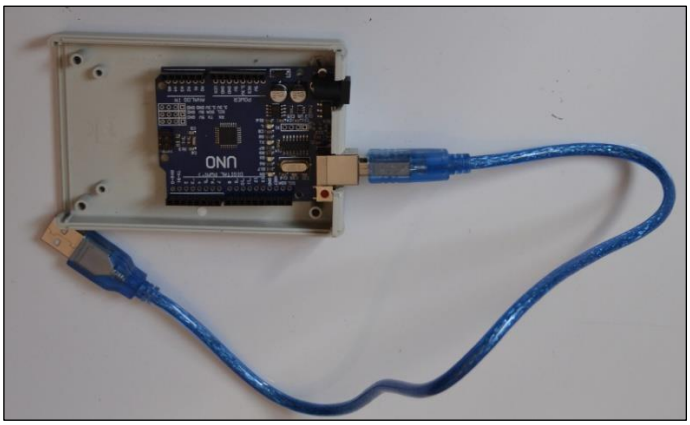

Figure 5.3 Arduino uno module (Personal archive, 2018)

5.1.4 RS232 Converter Module

The RS232 converter Module carries a Max232 chip that converts 115200 baud rates to 38400 baud rates. This means that the VC706 camera module works at 38400 baud rates - +5V and the HE910T GSM/GPS module works at 115200 baud rates – (+12V) and hence, it converts the camera data to a data format that the HE910T will read (Figure 5.4).

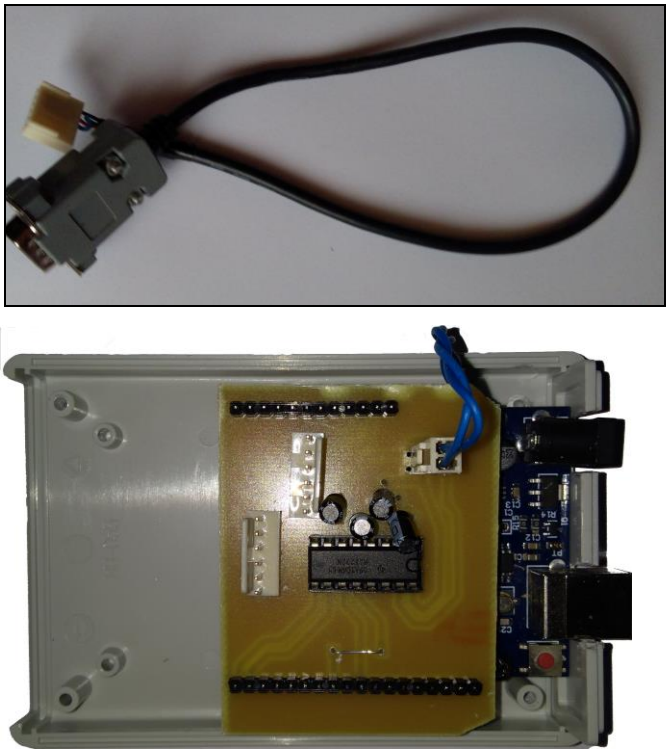

Figure 5.4 RS232 converter module (Personal archive, 2018)

5.2 The Working Mechanism of the Vehicle Kit

The main task of the vehicle kit is to send GPS and camera records to the web page, MySQL. One of its other tasks is to route the vehicle over the webpage.

Considering this aim, the vehicle kit's working mechanism can be easily explained using Figure 5.5 and Figure 5.6. First, the system begins with the HE910 module. The HE910 module receives GPS data from satellites and it keeps the GPS data in its memory to send to the MySQL database as HTTP protocol, every five seconds. Then, HTTP returns an answer against each sent data. The HE910 module sends a code to take a photo to Arduino Uno via the RS232 converter with 115200 baud rates, if the answer is 'takephoto***'. Arduino Uno sends back the photo data taken from the VC706 camera module with 38400 baud rates via the RS232 converter to the HE910 GSM/GPS module. If the answer is 'clear***', the HE910 module stops taking a photo.

Besides all these features, the vehicle kit has a USB output if it was necessary to see the vehicle kit's working mechanism programatically on a PC screen (Figure 5.8 and Figure 5.9). Developing the project is very easy using this feature.

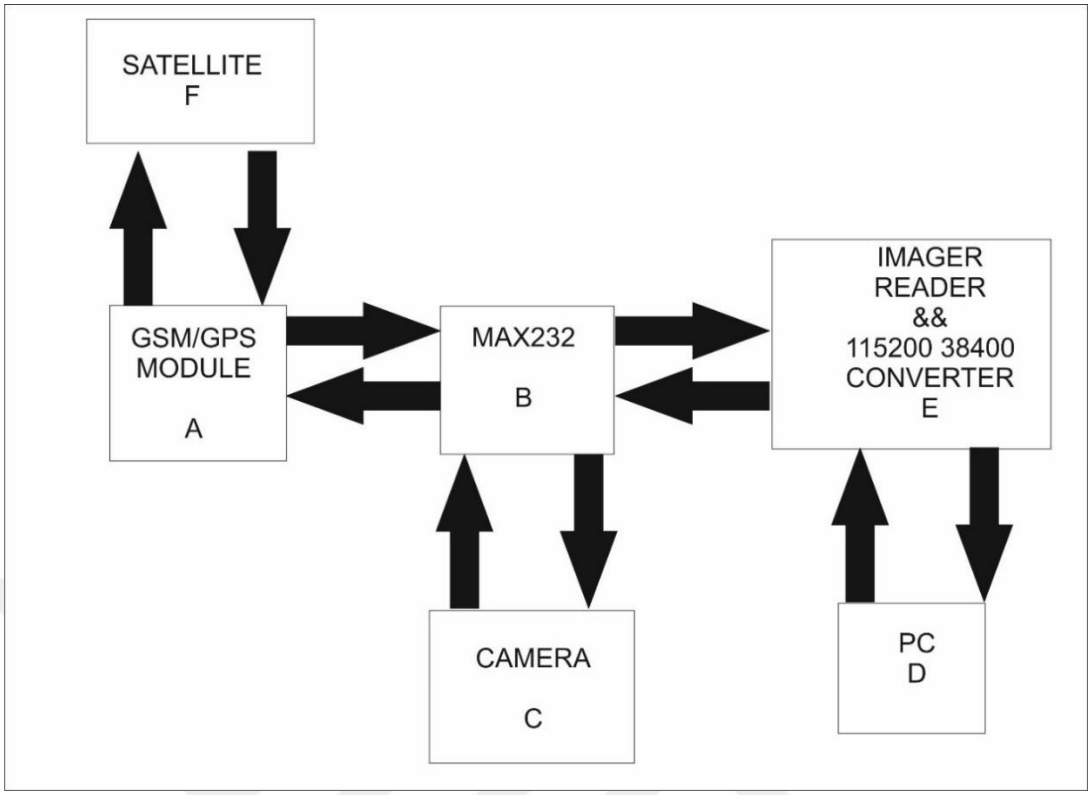

Figure 5.5 The working mechanism of the vehicle kit

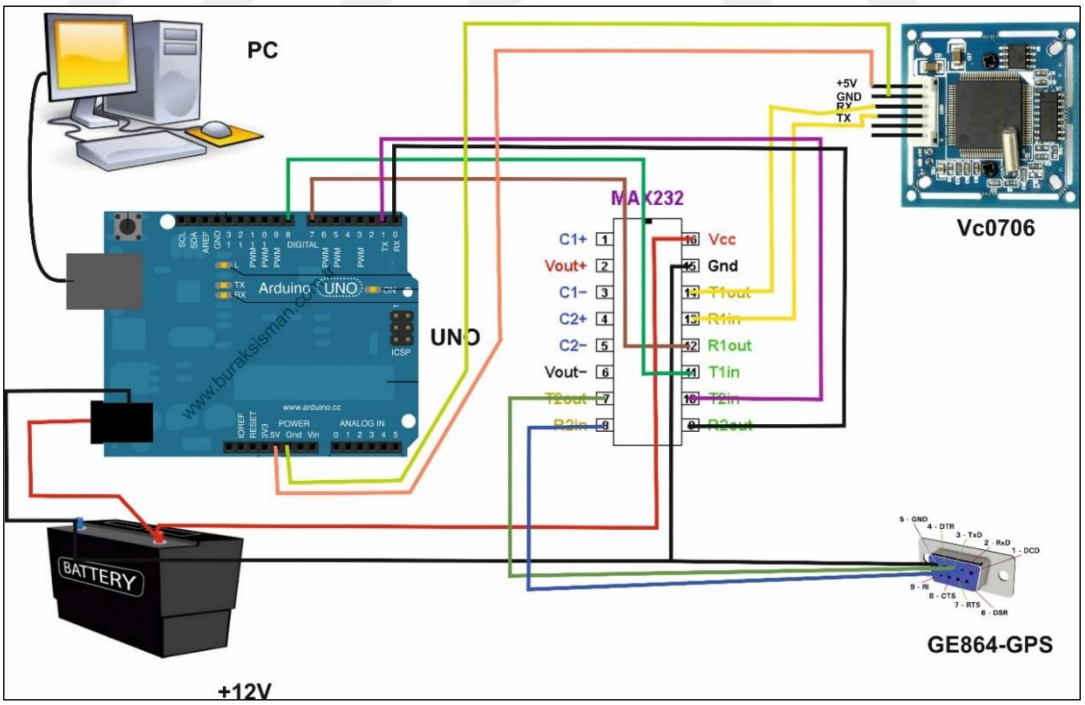

Figure 5.6 The illustration of the working mechanism of the vehicle kit

Figure 5.7 Vehicle kit (Personal archive, 2018)

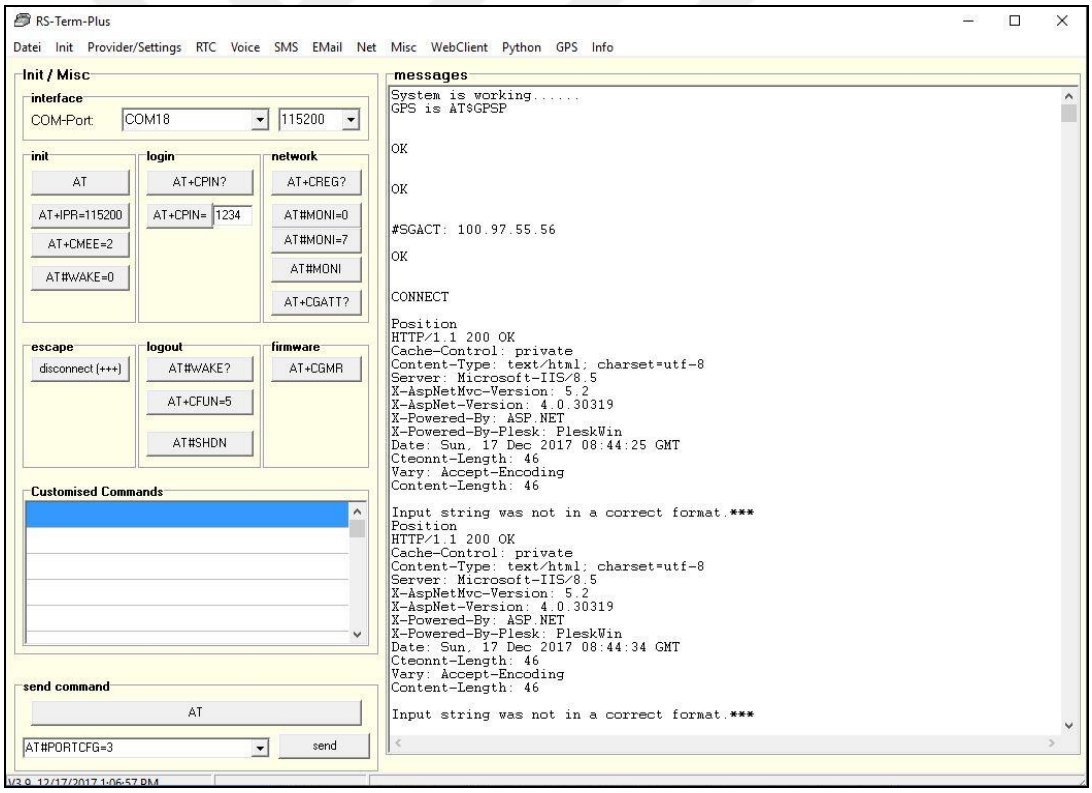

Figure 5.8 Monitoring the vehicle kit with PC-RealTerm[1]

[1] https://realterm.sourceforge.io/.

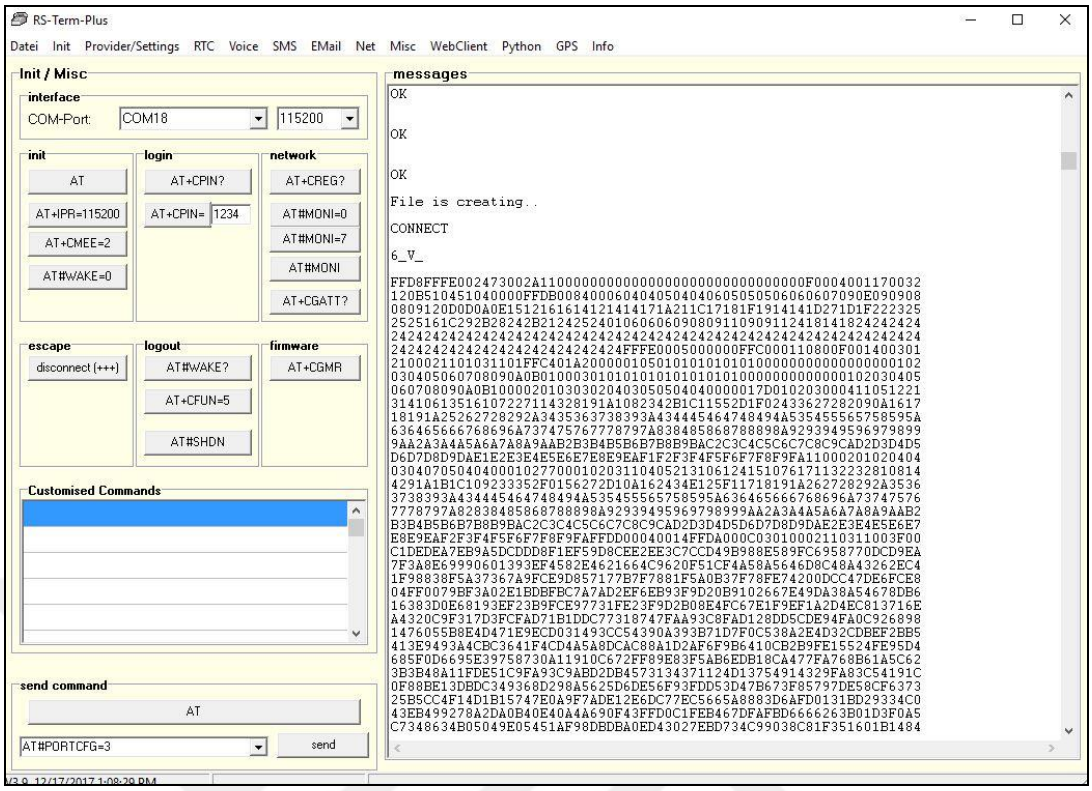

Figure 5.9 Monitoring the vehicle kit with PC-RealTerm

CHAPTER SIX
RESULTS AND CONCLUSION

6.1 Results

The system has accumulated approximately two-hundred thousand GPS records throughout the studies. The records indicate that the vehicle kit and the website perform their works reliable and standard deviation at a minimum risk level. The developable vehicle kit is fully comprehensive and economic device for corporations and individuals who demand to make their own vehicle tracking-routing system. MVC, Bootstrap, Google Map APIs and jQuery technologies implement to display the vehicle locations and camera images on the vehicle website despite different displaying devices (PC, mobile phone, tablet, iPad etc.).

Cameras have a deterrent ability against thieves. A camera that is placed inside a vehicle is considered to provide security against thieves. As well, camera images are an evidence.

The system is rather fast and stable enough to respond to MySQL adhoc queries. We must not forget that GMS communication is accepted at normal state.

6.2 Advantages of the Vehicle Tracking-Routing System

1. The vehicle is easily monitored remotely, via the webpage.
2. It is simple to take a photo inside the vehicle.
3. A command can be sent to the vehicle kit via the webpage for any purpose. The vehicle has digital voltage input and output pins. Users can shut off the engine fom far away by plugining the pins to the engine control unit of vehicle.
4. The vehicle is displayed on private maps in the GIS software using WMS.
5. Determined polygons as GIS can be used for various applications. The system sends an e-mail to the user when any event is triggered. For example, when the vehicle is inside the polygon or out of the polygon, an event can be triggered.
6. The vehicle kit is an open code software to encode and develop, because, the vehicle kit can be corresponded with a PC via the RS232 interface. Both the

HE910 and the Arduino have digital and analog input-output pins for any sensor and relays.

7. All codes can be found in the Appendices section.

6.3 Disadvantages of the Vehicle Tracking-Routing System

1. Dimensions of the vehicle kit should be shrunk, because, it would be easier to hide the vehicle kit in the vehicle.
2. The webpage database connection string and table querying should be converted to Language Integrated Query (LINQ). The encoding of the webpage will make it more convenient and more understandable.

REFERENCES

Anderson, R. (2014). Getting Started with ASP .NET MVC 5. *Viitattu*, *8*, 2015.

Aryal, S.C. (2014). *Design principles for responsive web*. Undergraduate Thesis, Helsinki Metropolia University, Helsinki.

ASP.NET Core 1.0 documentation release. (n.d.). Retrieved March 5, 2018, from https://media.readthedocs.org/pdf/aspnet/latest/aspnet.pdf.

Chadil, N., Russameesawang, A. & Keeratiwintakorn, P. (2008). Real-time tracking management system using GPS, GPRS and Google Earth. *ECTI-CON 2008. 5th International Conference on*. *Krabi-Thailand*, 393-396.

Davis, S. (2007). *GIS for web developers*. Texas: The Pragmatic Bookshelf.

Dincer, A., & Uraz, B. (2013). *Google Maps JavaScript API Cookbook*. Birmingham B3 2PB, UK: Packt Publishing Ltd.

Frain, B. (2015). *Responsive web design with HTML5 and CSS3*. Birmingham B3 2PB, UK: Packt Publishing Ltd.

Google (n.d.). Retrieved September 15, 2019, from https://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&callback=initMap"

HE910 family product description. (n.d.). Retrieved December 13, 2017, from https://www.telit.com/wp-content/uploads/2017/09/Telit_HE910_Product_Description_r11.pdf

Jovičić, B., & Simić, D. (2006). Common web application attack types and security using ASP .NET. *ComSIS*, *3*(2), 83-96.

Khoury, F., & Zgheib, A. (2018). *Building a dedicated GSM GPS module tracking system for fleet management: hardware and software*. London: CRC Press.

Kodavati, B., Raju, V. K., Rao, S. S., Prabu, A. V., Rao, T. A., & Narayana, D. Y. (2011). GSM and GPS based vehicle location and tracking system. *International Journal of Engineering Research and Applications (IJERA)*, *1*(3), 616-625.

Lee, S., Tewolde, G., & Kwon, J. (2014, March). Design and implementation of vehicle tracking system using GPS/GSM/GPRS technology and smartphone application. In *2014 IEEE World Forum on Internet of Things (WF-IoT)* 353-358.

LinkSprite Technologies. (n.d.). Retrieved Jun 12, 2017, from https://www.sparkfun.com/datasheets/Sensors/Imaging/1274419957.pdf.

Maurya, K., Singh, M., & Jain, N. (2012). Real time vehicle tracking system using GSM and GPS technology-an anti-theft tracking system. *International Journal of Electronics and Computer Science Engineering*, *1*(3), 1103-107.

McDonald, M. (2006). *Intelligent Transport Systems in Europe: opportunities for future research*. London, UK: World Scientific.

Norman, M. (2006). *Database design manual: using MysQL for windows*. London: Springer Science & Business Media.

Open Geospatial Consortium. (n.d.). Retrieved January 12, 2018, from http://www.opengeospatial.org/standards/wms/.

Stillwell, J., & Clarke, G. (Eds.). (2004). *Applied GIS and spatial analysis*. Chichester: Wiley.

Svennerberg, G. (2010). *Beginning Google Maps API 3*. New York: Apress.

The MapServer Team. (n.d.). Retrieved February 3, 2018, from http://download.osgeo.org/mapserver/docs/MapServer.pdf.

Torre, C., & Carmona, D. (2013). .*NET Technology Guide for Business Applications*. Redmond, Washington: Microsoft Press.

Verma, P., & Bhatia, J. S. (2013). Design and development of GPS-GSM based tracking system with Google map based monitoring. *International Journal of Computer Science, Engineering and Applications*, *3*(3), 33.

Zakas, N. C. (2009). *JavaScript for Web Developers*. Indianapolis, Indiana: John Wiley & Sons.

.

APPENDICES

Appendix 1: Web Page Codes-Visual Studio ASP NET MVC 5

Models Codes in MVC

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace VTRS.Models
{
    public class Polygons
    {
        public string PolygonName { get; set; }
        public string Email { get; set; }
        public virtual ICollection<Location> vetrex { get; set; }
        public int id { get; set; }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace VTRS.Models
{
    public class Location
    {
        public float lng { get; set; }
        public float lat { get; set; }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace VTRS.Models
{
    public class GPS
    {
        public float Lati { get; set; }
        public float Longi { get; set; }
        public float Speed { get; set; }
        public float EastingED50 { get; set; }
        public float NorthingED50 { get; set; }
        public int UTM { get; set; }
```

```
        public int id { get; set; }
        public string VehicleDate { get; set; }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace VTRS.Models
{
    public class PhotoInfo
    {
        public bool IsPhotoReady { get; set; }


    }
}
```

MapController Codes in MVC

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using MySql.Data.MySqlClient;
using System.IO;
using System.Net;
using System.Globalization;
using VTRS.Models;
using arfpiCoordinateConverter;
using System.Data;



namespace VTRS.Controllers
{


public class MapController : Controller
{

MySqlConnection                  cnn                  =                  new
MySqlConnection("Server=xx.xx.xx.xx;Database=xxxx;Uid=xxxxx;Pwd=xxxx;");
        // GET: Map
        public MapController()
        {
```

```
            CultureInfo                    customCulture                    =
(CultureInfo)System.Threading.Thread.CurrentThread.CurrentCulture.Clone();
            customCulture.NumberFormat.NumberDecimalSeparator = ".";
            System.Threading.Thread.CurrentThread.CurrentCulture            =
customCulture;
        }
        public ActionResult Index()
        {
            PhotoInfo IsphotoReady = new PhotoInfo();

            try
            {
                cnn.Open();
                MySqlDataAdapter  adapter1  =  new  MySqlDataAdapter("Select
Photo From mvcttakephoto", cnn);
                System.Data.DataSet ds1 = new System.Data.DataSet();
                adapter1.Fill(ds1);
                cnn.Close();
                if (ds1.Tables[0].Rows[0][0].ToString() == "clear***")
                    IsphotoReady.IsPhotoReady = true;
                else
                    IsphotoReady.IsPhotoReady = false;
                return View(IsphotoReady);
            }
            catch (Exception)
            {

                throw;
            }
            IsphotoReady.IsPhotoReady = false;
            return View(IsphotoReady);
        }


public ActionResult SavePolygon(string polygonn,string namee)
        {
            var query = @"insert into deupolygons (PolygonName,GIS) Values('"
+ namee + "',GeomFromText('" + polygonn + "'))";
            cnn.Open();
            MySqlCommand cmd = new MySqlCommand(query, cnn);
            cmd.ExecuteNonQuery();
            cnn.Close();
            return  Json(new  {  result  =  "'saving  Polygon  Geoemetry  '  is
successfull" }, JsonRequestBehavior.AllowGet);


        }
        public ActionResult SaveGPS(string photo,string psw,string lat, string
longg,string speed)
        {
            try
            {

                if (photo == "1")
                {
                    cnn.Open();
                    MySqlDataAdapter adapter1 = new MySqlDataAdapter("update
mvcttakephoto Set Photo='clear***'", cnn);
                    System.Data.DataSet ds1 = new System.Data.DataSet();
                    adapter1.Fill(ds1);
```

```
                    cnn.Close();
                }
                cnn.Open();
                MySqlDataAdapter adapter = new MySqlDataAdapter("Select Photo
From mvcttakephoto", cnn);
                System.Data.DataSet ds = new System.Data.DataSet();
                adapter.Fill(ds);
                cnn.Close();

                if (speed == null) speed = "0";
                if (speed.Length < 1) speed = "0";

                    Double lat_ = Convert.ToDouble(lat) / 100;
                    Double long_ = Convert.ToDouble(longg) / 100;
                    lat_ = (int)lat_ + ((lat_ - (int)lat_) * 100) / 60;
                    long_ = (int)long_ + ((long_ - (int)long_) * 100) / 60;
                    cnn.Open();
                    MySqlCommand cmd = new MySqlCommand("insert into deucarlog
(Latitude, Longitude,Plate,Speed,Date) Values(" +
                        float.Parse(lat_.ToString(),
System.Globalization.CultureInfo.InvariantCulture.NumberFormat) + "," +
                        float.Parse(long_.ToString(),
System.Globalization.CultureInfo.InvariantCulture.NumberFormat) + ",''," +
                        float.Parse(speed,
System.Globalization.CultureInfo.InvariantCulture.NumberFormat)  +  ",'"  +
DateTime.Now.ToString("yyyyMMdd HH:mm:ss") + "')", cnn);
                    cmd.ExecuteNonQuery();
                var _query = "select distinct p.Email from deucarlog c inner
join  deupolygons  p  on  MBRWithin(Point(c.Longitude,c.Latitude),p.GIS)  =1
limit 5";
                cnn.Close();

                cnn.Open();
                adapter   =   new   MySqlDataAdapter("Select   Photo   From
mvcttakephoto", cnn);
                ds = new System.Data.DataSet();
                adapter.Fill(ds);
                cnn.Close();
                foreach (DataRow item in ds.Tables[0].Rows)
                {
                    sendEmail(item[0].ToString());
                }
                return Content(ds.Tables[0].Rows[0][0].ToString() + "\r\n");
            }
            catch (Exception ex)
            {
                cnn.Close();
                return Content(ex.Message + "***\r\n");
            }
            return Content("it was saved.***" +"\r\n");
        }

public ActionResult getGPS(string plateid)
        {
            ICollection<GPS> gps = new List<GPS>();

            cnn.Open();
```

```
            MySqlDataAdapter adapter = new MySqlDataAdapter("Select Latitude,
Longitude, Speed, Date,EastingED50,NorthingED50,UTM,id From deucarlog where
speed>4 order by id desc limit 4000", cnn);
            System.Data.DataSet ds = new System.Data.DataSet();
            adapter.Fill(ds);
            cnn.Close();
            foreach (System.Data.DataRow item in ds.Tables[0].Rows)
            {
                gps.Add(new GPS()
                {
                    Lati     =     float.Parse(item[0].ToString().ToString(),
System.Globalization.CultureInfo.InvariantCulture.NumberFormat),
                    Longi    =     float.Parse(item[1].ToString().ToString(),
System.Globalization.CultureInfo.InvariantCulture.NumberFormat),
                    Speed    =     float.Parse(item[2].ToString().ToString(),
System.Globalization.CultureInfo.InvariantCulture.NumberFormat),
                    EastingED50 = float.Parse(item[4].ToString().ToString(),
System.Globalization.CultureInfo.InvariantCulture.NumberFormat),
                    NorthingED50 = float.Parse(item[5].ToString().ToString(),
System.Globalization.CultureInfo.InvariantCulture.NumberFormat),
                    UTM = Convert.ToInt32(item[6].ToString().ToString()),
                    id = Convert.ToInt32(item[7].ToString().ToString()),
                    VehicleDate = item[3].ToString().ToString()
                });
            }
            return Json(gps, JsonRequestBehavior.AllowGet);
        }

        public ActionResult getline(string plateid)
        {
            ICollection<GPS> gps = new List<GPS>();

            cnn.Open();
            MySqlDataAdapter adapter = new MySqlDataAdapter("Select Latitude,
Longitude From deucarlog where speed>4 order by id desc limit 4000", cnn);
            System.Data.DataSet ds = new System.Data.DataSet();
            adapter.Fill(ds);
            cnn.Close();
            ICollection<Location> location = new List<Location>();
            foreach (System.Data.DataRow item in ds.Tables[0].Rows)
            {
                location.Add(new Location()
                {
                    lat      =     float.Parse(item[1].ToString().ToString(),
System.Globalization.CultureInfo.InvariantCulture.NumberFormat),
                    lng      =     float.Parse(item[0].ToString().ToString(),
System.Globalization.CultureInfo.InvariantCulture.NumberFormat)
                });

            }
            return Json(location, JsonRequestBehavior.AllowGet);
        }


public ActionResult updatePoly(int row,string email)
        {
            var query = @"update deupolygons Set Email='" + email + "' where
id=" + row;
            cnn.Open();
            MySqlCommand cmd = new MySqlCommand(query, cnn);
```

```
            cmd.ExecuteNonQuery();
            cnn.Close();
            return Json(new { result = "'Updating e-mail ' is successfull" +
query }, JsonRequestBehavior.AllowGet);

        }
        public ActionResult getPolygon()
        {
            ICollection<GPS> gps = new List<GPS>();

            cnn.Open();
            MySqlDataAdapter    adapter    =    new    MySqlDataAdapter("select
Replace(Replace(Replace(AsText(GIS),'POLYGON((',''),'))',''),',   ',',')   As
Vertex,PolygonName,Id,Email from deupolygons", cnn);
            System.Data.DataSet ds = new System.Data.DataSet();
            adapter.Fill(ds);
            cnn.Close();
            ICollection<Polygons> polygons = new List<Polygons>();
            foreach (System.Data.DataRow item in ds.Tables[0].Rows)
            {
                List<Location> vertexx = new List<Location>();
                var _splt = item[0].ToString().Split(',');
                foreach (var _splts in _splt)
                {
                    var _coord = _splts.Split(' ');
                    vertexx.Add(new Location {
                        lat             =             float.Parse(_coord[0],
System.Globalization.CultureInfo.InvariantCulture.NumberFormat),
                        lng             =             float.Parse(_coord[1],
System.Globalization.CultureInfo.InvariantCulture.NumberFormat),
                    });
                }
                polygons.Add(new Polygons {
                    PolygonName = item[1].ToString(),
                    vetrex = vertexx,
                    id = Convert.ToInt32(item[2].ToString()),
                    Email    = item[3].ToString()
                });
            }

            return Json(polygons, JsonRequestBehavior.AllowGet);
        }
        public ActionResult GetPhoto()
        {

            return base.File(ReadImage(), "image/jpeg");
        }


public ActionResult TakePhoto()
        {
            try
            {

                cnn.Open();
                MySqlDataAdapter   adapter   =   new   MySqlDataAdapter("update
mvcttakephoto Set Photo='takephot***'", cnn);
                System.Data.DataSet ds = new System.Data.DataSet();
                adapter.Fill(ds);
                cnn.Close();
```

```
                return Json(new { result = "Successfull" }, JsonRequestBehavior.AllowGet);
            }
            catch (Exception)
            {
                cnn.Close();
            }
            return Json(new { result = "Error" }, JsonRequestBehavior.AllowGet);

        }

        public byte[] hexToByteArray(string hexString)
        {
            int bytesCount = (hexString.Length) / 2;
            byte[] bytes = new byte[bytesCount];
            for (int x = 0; x < bytesCount; ++x)
            {
                bytes[x] = Convert.ToByte(hexString.Substring(x * 2, 2), 16);
            }
            //return byte array
            return bytes;
        }

        private byte[] ReadImage()
        {
            try
            {
                FtpWebRequest request = (FtpWebRequest)WebRequest.Create("ftp://xx.xx.xx.xx/vtr.txt");
                request.Method = WebRequestMethods.Ftp.DownloadFile;
                request.Credentials = new NetworkCredential("xxxxxx", "xxxxx");
                FtpWebResponse response = (FtpWebResponse)request.GetResponse();
                Stream responseStream = response.GetResponseStream();

                StreamReader reader = new StreamReader(responseStream);
                string message = "";
                string hexchar = reader.ReadToEnd().Replace(" ", "").Replace(Environment.NewLine, "").Replace("6_", "").Replace("_", "").Replace("V", "");
                if (hexchar.IndexOf("FFD8") < 0)
                    message = "FFD8" + hexchar;
                else
                    message = hexchar;
                byte[] imageByte =hexToByteArray(message);
                reader.Close();
                response.Close();
                return imageByte;
            }
            catch
            {

                return null;
            }
        }

        public ActionResult CalculateED50()
        {
```

```
            try
            {
                cnn.Open();
                MySqlDataAdapter adapter = new MySqlDataAdapter("SELECT
l.Latitude, l.Longitude, l.Id, m.UTM FROM deucarlog l INNER JOIN  deuutm m ON
MBRWithin(Point(l.Latitude,l.Longitude), m.GIS) = 1 where l.EastingED50  is
NULL order by id", cnn);
                DataSet ds = new DataSet();
                adapter.Fill(ds);
                cnn.Close();
                var func = new sysCoordinate();
                cnn.Open();
                foreach (DataRow item in ds.Tables[0].Rows)
                {
                    string x, y;
                    x = "0";
                    y = "0";

                    if (item[3].ToString() == "38")
                    {
                        var                         res                         =
func.convertCoordinateFromTo(item[0].ToString(), item[1].ToString(), "4230",
"23038");
                        var splt= res.Split('|');
                        x = splt[0];
                        y = splt[1];
                    }
                    else if (item[3].ToString() == "37")
                    {
                        var                         res                         =
func.convertCoordinateFromTo(item[0].ToString(), item[1].ToString(), "4230",
"23037");
                        var splt = res.Split('|');
                        x = splt[0];
                        y = splt[1];
                    }
                    else if (item[3].ToString() == "36")
                    {
                        var                         res                         =
func.convertCoordinateFromTo(item[0].ToString(), item[1].ToString(), "4230",
"23036");
                        var splt = res.Split('|');
                        x = splt[0];
                        y = splt[1];
                    }
                    else if (item[3].ToString() == "35")
                    {
                        var                         res                         =
func.convertCoordinateFromTo(item[0].ToString(), item[1].ToString(), "4230",
"23035");
                        var splt = res.Split('|');
                        x = splt[0];
                        y = splt[1];
                    }

                    MySqlCommand cmd = new MySqlCommand("Update deucarlog Set
EastingED50="  + x + ",NorthingED50=" + y  + ",UTM=" + item[3].ToString() + "
where id=" + item[2].ToString(),cnn);
                    cmd.ExecuteNonQuery();
                }
```

```
                cnn.Close();
            }
            catch (Exception)
            {
                cnn.Close();
                return Json(new { result = "Error" },
JsonRequestBehavior.AllowGet);
            }


            return Json(new { result = "'Calculate Geographic WGS84 to ED50'
is successfull" }, JsonRequestBehavior.AllowGet);

        }

        public ActionResult CalculateSpeed(int id1,int id2)
        {
            try
            {
                cnn.Open();
                MySqlDataAdapter adapter = new MySqlDataAdapter("SELECT (Max(
TIMESTAMPDIFF(SECOND, STR_TO_DATE('20170901 17:00:00','%Y%m%d
%H:%i:%s'),STR_TO_DATE(Date,'%Y%m%d %H:%i:%s')))- Min( TIMESTAMPDIFF(SECOND,
STR_TO_DATE('20170901 17:00:00','%Y%m%d %H:%i:%s'),STR_TO_DATE(Date,'%Y%m%d
%H:%i:%s'))))/3600 FROM deucarlog where Date is not null and EastingED50 is
not null and id in (" + id1 + "," + id2 + ")", cnn);
                DataSet ds = new DataSet();
                adapter.Fill(ds);
                cnn.Close();
                double _hour =
Convert.ToDouble(ds.Tables[0].Rows[0][0].ToString());
                if (_hour == 0)
                    _hour = 0.001388889;
                cnn.Open();
                adapter = new MySqlDataAdapter("SELECT Latitude,Longitude
FROM deucarlog where Date is not null and EastingED50 is not null and id in
(" + id1 + "," + id2 + ")", cnn);
                ds = new DataSet();
                adapter.Fill(ds);
                cnn.Close();
                var _distance = new arfpiFunctions();
                var lat1 =
Convert.ToDouble(ds.Tables[0].Rows[0][0].ToString());
                var long1 =
Convert.ToDouble(ds.Tables[0].Rows[0][1].ToString());
                var lat2 =
Convert.ToDouble(ds.Tables[0].Rows[1][0].ToString());
                var long2 =
Convert.ToDouble(ds.Tables[0].Rows[1][1].ToString());
                var _km = _distance.getDistanceFromLatLonInKm(lat1, long1,
lat2, long2);
                var _speed = _km / _hour;
                return Json(new { result = _speed.ToString() },
JsonRequestBehavior.AllowGet);
            }
            catch (Exception)
            {
                cnn.Close();
            }
```

```
            return      Json(new      {      result      =      "Null"      },
JsonRequestBehavior.AllowGet);
        }

        public void sendEmail(string email)
        {
            string mailaddress = "ozkank@arfpi.net";
            int port = 587;
            string server = "xxxxxx.com.tr";
            string password = "xxxxxx";
            try
            {
                System.Net.Mail.MailMessage         msg         =         new
System.Net.Mail.MailMessage(mailaddress, email);
                msg.Subject = "DEU Vehicle Tracking System";
                msg.Body = "There is a vehicle in polygon which was drawn.";
                msg.SubjectEncoding                                          =
System.Text.Encoding.GetEncoding("windows-1254");
                msg.BodyEncoding = System.Text.Encoding.GetEncoding("windows-
1254");
                msg.IsBodyHtml = true;
                System.Net.Mail.SmtpClient         smt         =         new
System.Net.Mail.SmtpClient(server, port);
                smt.Credentials                      =                    new
System.Net.NetworkCredential(mailaddress, password);
                smt.DeliveryMethod                                           =
System.Net.Mail.SmtpDeliveryMethod.Network;
                smt.Credentials                      =                    new
System.Net.NetworkCredential(mailaddress, password);
                smt.EnableSsl = true;
                smt.Send(msg);
                smt.Dispose();


            }
            catch { }
        }
    }
}
```

Index View codes in MVS

```
@model VTRS.Models.PhotoInfo
@{
    ViewBag.Title = "DEU Vehicle Tracking and Routing System";
}
<link        rel="stylesheet"        href="http://fontawesome.io/assets/font-
awesome/css/font-awesome.min.css" />
<link                 href="https://fonts.googleapis.com/css?family=Lobster"
rel="stylesheet">
```

```
<style>
    html{
    height: 100vh;
    margin:0px;
    padding:0px;
    width:100%;
    }
    body {
        /*background-color: #0275D8;*/
        background-color: #ffffff;
        width:100%;
    }

    h2 {
        font-family:  'Lobster','Franklin  Gothic  Medium',  'Arial  Narrow',
Arial, sans-serif;
        text-align: center;
        color: #0275D8;
    }
    .map{
        height:100vh;
        width:100%;
        margin-left:-20px;
        top:0;
        position:absolute;
        z-index:0;
    }

    .mapMenu{
        margin-top:150px;
    }
    .row{
       position:absolute;
       top:100px;
       margin: 0 auto;
       z-index:1;
    }
    .row div{
        background-color:powderblue;
        color:white;
        padding:2px;
        width:80px;
    }
    .large-image{
        width: 100%;
        height: auto;
        display: block;

    }

</style>



<nav class="navbar fixed-top navbar-expand-lg navbar-light bg-light">
    <img  src="~/images/logo_deu.png"  class="img-thumbnail"  width="50"><img
class="img-thumbnail" src="~/images/fbe.jpg"  width="50"/>
```

```
    <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-
expanded="false" aria-label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
    </button>

    <div class="collapse navbar-collapse" id="navbarSupportedContent">
        <ul class="navbar-nav mr-auto">
            <li class="nav-item">
                <a class="nav-link" href="#" id="getgps">Show Waypoints</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="#" id="getline">Show Line</a>
            </li>
            <li class="nav-item">
                <a    class="nav-link"    href="#"    id="tediousTraffic">Show
Traffic</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="#" id="takephoto">Take a Photo</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="#" id="showphoto">Show a Photo</a>
            </li>
            <li class="nav-item dropdown">
                <a      class="nav-link      dropdown-toggle"      href="#"
id="navbarDropdown" role="button" data-toggle="dropdown" aria-haspopup="true"
aria-expanded="false">
                    Basemap
                </a>
                <div class="dropdown-menu" aria-labelledby="navbarDropdown">
                    <a class="dropdown-item" href="#" id="basemap1">ROAD</a>
                    <a             class="dropdown-item"             href="#"
id="basemap2">SATELLITE</a>
                    <div class="dropdown-divider"></div>
                    <a             class="dropdown-item"             href="#"
id="basemap3">HYBRID</a>
                </div>
            </li>
            <li class="nav-item dropdown">
                <a      class="nav-link      dropdown-toggle"      href="#"
id="navbarDropdownCalculation"  role="button"  data-toggle="dropdown"  aria-
haspopup="true" aria-expanded="false">
                    Calculation
                </a>
                <div               class="dropdown-menu"               aria-
labelledby="navbarDropdownCalculation">
                    <a             class="dropdown-item"             href="#"
id="calcWGS84toED50">Calculate Geographic WGS84 to ED50</a>
                    <a             class="dropdown-item"             href="#"
id="calcSpeed">Calculate Speed</a>
                    <a class="dropdown-item" href="#" id="drawPolygon">Draw a
Polygon</a>
                    <a             class="dropdown-item"             href="#"
id="QueryInPolygon">Querying Vehicle in Polygon</a>
                    <a  class="dropdown-item"  href="#"  id="showPolygon">Show
Polygons</a>
                </div>
            </li>
            <li class="nav-item dropdown">
```

```
                <a       class="nav-link       dropdown-toggle"       href="#"
id="navbarDropdownHelp"     role="button"     data-toggle="dropdown"     aria-
haspopup="true" aria-expanded="false">
                    Help
                </a>
                <div               class="dropdown-menu"               aria-
labelledby="navbarDropdownHelp">
                    <a  class="dropdown-item"  href="#"  id="wmslink"  data-
toggle="modal" data-target="#exampleModal">How to Connect WMS Server</a>

                </div>
            </li>
        </ul>
        <form class="form-inline my-2 my-lg-0">
            <input      class="form-control      mr-sm-2"      type="search"
placeholder="Search" aria-label="Search">
            <button    class="btn    btn-outline-success    my-2    my-sm-0"
type="submit">Search</button>
        </form>
    </div>
</nav>



<div class="container">
    <br />
    @if (Model.IsPhotoReady == true)
    {
        <div class="row" id="imageBox">
            <img             class="img-thumbnail             large-image"
src="@Url.Action("GetPhoto","Map")" /><div id="imageClose">Close</div>
        </div>
    }
    else
    {
        <div class="row">
            <div     class="mapImage">Photo     is     preparing.     Please,
waitting...</div>
        </div>
    }

</div>

<div class="row mapMenu" id="calcspeed_1"><h2>DEU Vehicle Tracking and Routing
System</h2></div>
<div class="map" id="map"></div>



<div class="modal fade" id="exampleModal" tabindex="-1" role="dialog" aria-
labelledby="exampleModalLabel" aria-hidden="true">
    <div class="modal-dialog modal-lg" role="document">
        <div class="modal-content">
            <div class="modal-header">
                <h5  class="modal-title"  id="exampleModalLabel">WMS  Client
Link</h5>
```

```
                <button   type="button"   class="close"   data-dismiss="modal"
aria-label="Close">
                    <span aria-hidden="true">&times;</span>
                </button>
            </div>
            <div class="modal-body">
                <p class="">
                    http://localhost/cgi-
bin/mapserv.exe?map=/ms4w/apps/mapserver/DEU_CAR.map&version=1.1.1&service=W
MS&request=GetMap&SRS=EPSG:4326&
                </p>
            </div>
            <div class="modal-footer">
                <button   type="button"   class="btn   btn-secondary"   data-
dismiss="modal">Close</button>
            </div>
        </div>
    </div>
</div>



<div class="modal fade" id="polygonModal" tabindex="-1" role="dialog" aria-
labelledby="polygonModalLabel" aria-hidden="true">
    <div class="modal-dialog modal-lg" role="document">
        <div class="modal-content">
            <div class="modal-header">
                <h5      class="modal-title"      id="polygonModalLabel">Add
Polygon</h5>
                <button   type="button"   class="close"   data-dismiss="modal"
aria-label="Close">
                    <span aria-hidden="true">&times;</span>
                </button>
            </div>
            <div class="modal-body">
               <label>Polygon Name</label><br />
                <input type="text" id="polygonName"/>
                <input type="hidden" id="polygonstring" />
            </div>
            <div class="modal-footer">
                <button   type="button"   class="btn   btn-secondary"   data-
dismiss="modal">Close</button>
                <button     type="button"     class="btn     btn-secondary"
id="polygonsave">Save</button>

            </div>
        </div>
    </div>
</div>



<div  class="modal  fade"  id="polygonEditModal"  tabindex="-1"  role="dialog"
aria-labelledby="polygonEdiModalLabel" aria-hidden="true">
    <div class="modal-dialog modal-lg" role="document">
        <div class="modal-content">
            <div class="modal-header">
```

```
                <h5   class="modal-title"   id="polygonEdiModalLabel">Polygon
Edit</h5>
                <button   type="button"   class="close"   data-dismiss="modal"
aria-label="Close">
                    <span aria-hidden="true">&times;</span>
                </button>
            </div>
            <div class="modal-body">
                <label>Polygon Name</label><br />
                <input  type="text"  readonly="readonly"  id="getpolygonName"
/><br />
                <label>Email Address</label><br />
                <input type="text"  id="emailName" /><button id="updateemail"
class="btn btn-warning">Update</button><br /><br />
                <button       id="deletepolygon"       class="btn       btn-
danger">Delete</button>
                <input type="hidden" readonly="readonly" id="getpolyid" /><br
/>

            </div>
            <div class="modal-footer">
                <button   type="button"   class="btn   btn-secondary"   data-
dismiss="modal">Close</button>
            </div>
        </div>
    </div>
</div>




<script
src="http://maps.googleapis.com/maps/api/js?key=xxxxxxxxxx&libraries=drawing
"></script>
<script src="https://code.jquery.com/jquery-3.2.1.min.js"></script>
@*<script             src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
integrity="sha384-
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
crossorigin="anonymous"></script>*@
<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.3/umd/popper.min.
js"                                                    integrity="sha384-
vFJXuSJphROIrBnz7yo7oB41mKfc8JzQZiCq4NCceLEaO4IHwicKwpJf9c9IpFgh"
crossorigin="anonymous"></script>
<script                src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-
beta.2/js/bootstrap.min.js"                                    integrity="sha384-
alpBpkh1PFOepccYVYDB4do5UnbKysX5WZXm3XxPqe5iKTfUKjNkCk9SaVuEZflJ"
crossorigin="anonymous"></script>




<script>

      var _lat = 39.930109176569594;
      var _long = 32.86611517944335;
      var _zoom =5;
      var mapProp = {
```

```
            center: new google.maps.LatLng(_lat, _long),
            zoom: _zoom,
            disableDefaultUI: true,
            mapTypeId: google.maps.MapTypeId.ROADMAP
        };
        var markers = [];
        var map = new google.maps.Map(document.getElementById("map"), mapProp);

        var takephoto = $("#takephoto");
        takephoto.click(function () {
                var http = $.ajax({
                    method: "GET",
                    url: "@Url.Action("TakePhoto", "Map")",
                    dataType: "json",
                    cache:false
                });
                http.done(function(item){
                    alert(item.result.toString())
                }).fail(function (obj) {
                    alert("Error:" + obj.result.toString());
                });
        });

          var getgps = $("#getgps");
          getgps.click(function () {
              for (var i=0;i<markers.length;i++){
                  markers[i].setMap(null);
              }
              markers = [];

              var http = $.ajax({
                  method: "GET",
                  url: "@Url.Action("getGPS","Map")",
                  dataType: "json",
                  cache:false,
              data:{plateid:"1"}
              });
              http.done(function (obj) {

                  $.each(obj, function (i, item) {

                      var   myCenter   =   new   google.maps.LatLng(item.Longi,
item.Lati);
                      var iconn = "point2.png";
                      if (i == 0) iconn = "point1.png";
                      // lastlocation = myCenter;
                     markers[i] = new google.maps.Marker({
                          position: myCenter,
                          icon: "http://www.arfpi.net/images/car/" + iconn
                      });
                      markers[i].setMap(map);
                      markers[i].addListener("click", function (event) {
                          var info = new google.maps.InfoWindow();
                          var _speed = "0";
                          var subhttp = $.ajax({
                              method: "GET",
                              url: "@Url.Action("CalculateSpeed", "Map")",
                              dataType: "json",
                              cache:false,
                          data:{id1:item.id,id2:item.id-1}
```

```
                        });
                        subhttp.done(function (subitem) {
                            _speed = subitem.result.toString();
                            info.setContent("<div>Id:  "  +  item.id  +
"</div><div>Date:  "  +  item.VehicleDate  +  "</div><div>Vehicle  speed:  "  +
item.Speed
                            +  "</div><div>ED50  UTM  Easthing:  "  +
item.EastingED50 + "</div><div>ED50 UTM Northing: "
                            + item.NorthingED50 + "</div><div>UTM: " + item.UTM
+ "</div>" + "<div class=\"btn btn-warning\" id=\"calcspeed\">Real Speed " +
_speed + "</div>");
                            info.setPosition(myCenter);
                            info.open(map);
                        }).fail(function (subobj) {
                            alert("Error:" + subobj.toString());
                        });

                    });
                });

            }).fail(function (obj) {
                alert("Error:" + obj.toString());
            });
        });

var lineSymbol = {
            path: google.maps.SymbolPath.BACKWARD_CLOSED_ARROW
        };

        var getline = $("#getline");
        getline.click(function () {
                var http = $.ajax({
                method: "GET",
                url: "@Url.Action("getline", "Map")",
                dataType: "json",
                cache:false,
                data:{plateid:"1"}
                });
            http.done(function (obj) {
                //alert(obj[0][0].toString());
                  var lines = new google.maps.Polyline({
                        path: obj,
                        icons: [{
                            icon: lineSymbol,
                            offset: '0%',
                            repeat: '60px'
                        }],
                        geodesic: true,
                        strokeColor: '#FF0000',
                        strokeOpacity: 1.0,
                        strokeWeight: 3
                    });
                    lines.setMap(map);
                }).fail(function (obj) {
                    alert("Error:" + obj.toString());
                });
        });

        var trafficLayer = new google.maps.TrafficLayer();
```

```
        var tediousTraffic = $("#tediousTraffic");
        tediousTraffic.click(function () {
            trafficLayer.setMap(map);
        });

        var basemap = $("[id*='basemap']");
        basemap.click(function () {
            if (this.id.toString() == "basemap1")
                map.setMapTypeId(google.maps.MapTypeId.ROADMAP);
            if (this.id.toString() == "basemap2")
                map.setMapTypeId(google.maps.MapTypeId.SATELLITE);
            if (this.id.toString() == "basemap3")
                map.setMapTypeId(google.maps.MapTypeId.HYBRID);
        });

        var imageClose = $("#imageClose");
        var imageBox = $("#imageBox");
        imageClose.click(function () {
            imageBox.css("visibility", "hidden");
        });
        imageClose.mousemove(function    ()    {    imageClose.css("cursor",
"pointer"); });

        var showphoto = $("#showphoto");
        showphoto.click(function () { imageBox.css("visibility", "visible");
});
        imageBox.css("visibility", "hidden");

        var calcWGS84toED50 = $("#calcWGS84toED50");
        calcWGS84toED50.click(function () {
                var http = $.ajax({
                    method: "GET",
                    url: "@Url.Action("CalculateED50", "Map")",
                    dataType: "json",
                    cache:false
                });
                http.done(function (item) {
                    alert(item.result.toString())
                }).fail(function (obj) {
                    alert("Error:" + obj.toString());
                });
        });
    ////DRAWING SECTION

    var drawPolygon = $("#drawPolygon");
    drawPolygon.click(function () {
        drawingManager.setMap(map);

    });

    var drawingManager = new google.maps.drawing.DrawingManager({
        drawingMode: google.maps.drawing.OverlayType.POLYGON,
        drawingControl: true,
        drawingControlOptions: {
            position: google.maps.ControlPosition.BOTTOM_CENTER,
            drawingModes: ['polygon']
        },
        markerOptions:                          {                          icon:
'https://developers.google.com/maps/documentation/javascript/examples/full/i
mages/beachflag.png' },
```

```
        circleOptions: {
            fillColor: '#ffff00',
            fillOpacity: 1,
            strokeWeight: 5,
            clickable: false,
            editable: true,
            zIndex: 1
        }
    });
    google.maps.event.addListener(drawingManager, 'overlaycomplete', function
(event) {
        var vertices = event.overlay.getPath();
        var contentString = 'POLYGON((';
        var fx, fy, lx, ly;
        fx = 0; fy = 0
        lx = 0; ly = 0;
        for (var i = 0; i < vertices.getLength() ; i++) {
            var xy = vertices.getAt(i);
            if (i == 0) {
                fx = xy.lat(); fy = xy.lng();
            }
            lx = xy.lat(); ly = xy.lng();
            contentString += xy.lat() + ' ' + xy.lng() + ', ';
        }
        if (lx != fx && ly != fy) {
            contentString += fx.toString() + ' ' + fy.toString();
        } else
            contentString = contentString.substring(0, contentString.length -
2);
        contentString += '))';
        $("#polygonstring").val(contentString);
        $('#polygonModal').modal();



var polygonsave = $("#polygonsave");
    polygonsave.click(function () {
        var poly = $("#polygonstring").val();
        var nam = $("#polygonName").val();
        var http = $.ajax({
            method: "GET",
            url: "@Url.Action("SavePolygon", "Map")",
            dataType: "json",
            cache: false,
            data: { polygonn: poly, namee: nam }
        });
            http.done(function (item) {
                alert(item.result.toString());
            }).fail(function (obj) {
                alert("Error:" + obj.toString());
            });
        });
        drawingManager.setMap(null);
    });
    var poly=[];
    var showPolygon = $("#showPolygon");
    showPolygon.click(function () {
        poly=[];
       var http = $.ajax({
```

```
                method: "GET",
                url: "@Url.Action("getPolygon", "Map")",
                dataType: "json",
                cache:false
        });
        http.done(function (obj) {
            $.each(obj, function (i, item) {
                    //alert(item.vetrex.toString());
                     poly[i] = new google.maps.Polygon({
                        path: item.vetrex,
                        strokeColor: '#FF0000',
                        strokeOpacity: 0.8,
                        strokeWeight: 3,
                        fillColor: '#FF0000',
                        fillOpacity: 0.35
                     });
                     poly[i].addListener('click', function () {
                         $("#getpolygonName").val(item.PolygonName);
                         $("#getpolyid").val(item.id);
                         $("#emailName").val(item.Email);

                         $('#polygonEditModal').modal();
                     });

                     poly[i].setMap(map);
            });
        }).fail(function (obj) {
            alert("Error:" + obj.toString());
        });


    });

    var updateemail = $("#updateemail");
    updateemail.click(function () {
        var id = $("#getpolyid").val();
        var emaill = $("#emailName").val();
        var http = $.ajax({
            method: "GET",
            url: "@Url.Action("updatePoly", "Map")",
            dataType: "json",
            cache: false,
            data: { row: id, email: emaill }
        });
            http.done(function (item) {
            alert(item.result.toString());
        }).fail(function (obj) {
            alert("Error:" + obj.toString());
        });
    });
</script>
```

Appendix 2: WMS Codes-MapServer

DEU_CAR.MAP

```
MAP
 NAME "DEU_CAR"
 STATUS ON
 SIZE 1200 900
 EXTENT 24 36 50 40
 UNITS DD
 SHAPEPATH "data"
 IMAGECOLOR 255 255 128
 FONTSET "../etc/fonts.txt"
 PROJECTION
  "init=EPSG:4326"
 End
WEB
METADATA
                    "wms_title" "DEU_CAR"
                    "wms_onlineresource"              "http://127.0.0.1/cgi-
bin/mapserv.exe?map=/ms4w/apps/mapserver/DEU_CAR.map&"
                    "wms_srs" "EPSG:4326"
                    "wms_enable_request" "*"
               End
End

LAYER
  NAME "DEU_CAR"
  TYPE POINT
  STATUS DEFAULT

  CONNECTIONTYPE OGR
  CONNECTION
"MYSQL:dbarfpi,host=xx.xx.xx.xx,user=xxxxx,password=xxxxx,port=3306"
  #CONNECTION
  DATA             "SELECT             Point(Latitude,Longitude)             As
Shape_Point,id,Substring(Date,1,10)  as  Info   from  deucarlog  where  speed>3
order by id desc limit 2000"
  LABELITEM "Info"
  METADATA
    "wms_title" "DEU_CAR"
  End

Class
    NAME "DEU_CAR"
    STYLE
        COLOR 255 0 0
        SIZE 24
        SYMBOL "point1.png"
    END


    LABEL
     COLOR  0 0 0
     FONT sans
     TYPE truetype
     SIZE 8
     POSITION AUTO
```

```
     PARTIALS FALSE
     OUTLINECOLOR 255 255 255
    END
  END
End # layer


End #MAP
```

Appendix 3: The Vehicle Kit Codes – Python 2.7

```
import SER
import MDM
import time
import GPS

answer = 0
loop = 1
errorrepeat = 0
stepbystep = 0
lastcommand = ''
sendtime = 0
task = 0

CR = '\r'
LF = '\n'
CRLF = CR + LF
SER.set_speed("115200","8N1")
SER.send("System is working......" + CRLF)

def sendAT(message):
    a = MDM.send(message + CRLF, sendtime)

def sendAT2(message):
    a = MDM.send(message + CRLF, 0)
    b = MDM.sendbyte(0x0D,0)
    c = MDM.sendbyte(0x0A,0)

def answer():
    res=''
    start = time.time()
    nogo = 1
    while (time.time()-start < 10 and nogo == 1):
        res = res + MDM.read()
        if (res.find("OK") > -1):
            nogo = 0
        if (res.find("CONNE") > -1):
            nogo = 0
        if (res.find("ERRO") > -1):
            nogo = 0
    if (res.find("NO CARR") > -1):
        stepbystep = 2
    return res

def answer2():
    res=''
    start = time.time()
```

```
    nogo = 1
    while (time.time()-start < 10 and nogo == 1):
        res = res + MDM.read()
        if (res.find("***") > -1):
            nogo = 0
    if (res.find("NO CARR") > -1):
        stepbystep = 2
    return res

def ATCommand():
    errorrepeat = 0
    sendAT(lastcommand)
    res = answer()
    while (res.find("ERROR")>-1 and errorrepeat < 3):
        errorrepeat = errorrepeat + 1
        time.sleep(1)
        sendAT(lastcommand)
        res = answer()
        SER.send(res + CRLF)
    SER.send(res + CRLF)
    sendtime = 0
    time.sleep(1)
    return res

def GPS_Config():
    status = GPS.getPowerOnOff()
    if status == 0:
        GPS.powerOnOff(1)
        SER.send("GPS is AT$GPSP" + CRLF)
        lastcommand = "AT$GPSAT=1"
        res = ATCommand()

def getGPSLocation():
    try:
        GPSSTRING = GPS.getLastRMC()
        GPSSTRING = GPSSTRING.strip()
        if (GPSSTRING.find(',') != -1):
            posList = GPSSTRING.split(',')
            url = "lat=" + posList[5] + "longg=" + posList[3] + "speed=" +
posList[7]
            return posList[5] + "-" + posList[3] + "-" + posList[7] + "-" +
url
        else:
            return "0-0-0-0"
    except:
        return "0-0-0-0"

GPS_Config()

while loop == 1:
    if (stepbystep == 0):
        task = 0
        lastcommand = "AT"
        res = ATCommand()
        if (res.find("ERROR") > -1):
            stepbystep = 0
        else:
            stepbystep = 1

    if (stepbystep == 1):
```

```
        lastcommand = "AT+CGDCONT=1,""IP"",""internet"""
        res = ATCommand()
        if (res.find("ERROR") > -1):
            stepbystep = 0
        else:
            stepbystep = 2

    if (stepbystep == 2):
        lastcommand = "AT#SGACT=1,1"
        res = ATCommand()
        if (res.find("ERROR") > -1):
            stepbystep = 5
        else:
            stepbystep = 3

    if (stepbystep == 200):
        lastcommand = "AT#SGACT=1,1"
        res = ATCommand()
        if (res.find("ERROR") > -1):
            stepbystep = 205
        else:
            stepbystep = 6

    if (stepbystep == 3):
        lastcommand = "AT#SKTD=0,80,""www.arfpi.net"",0"
        res = ATCommand()
        if (res.find("ERROR") > -1):
            stepbystep = 0
        else:
            stepbystep = 4

    if (stepbystep == 4):
        coordinate = getGPSLocation()
        coordinate = coordinate.split('-')
        SER.send('Position' + coordinate[0] + CRLF)
        if (task == 1):
            sendAT("GET
http://deu.arfpi.net/Map/SaveGPS?photo=1&psw=xxxx&lat="  +  coordinate[0]  +
"&longg=" + coordinate[1] + "&speed=" + coordinate[2] + " HTTP/1.1")
        else :
            sendAT("GET
http://deu.arfpi.net/Map/SaveGPS?photo=0&psw=xxxx&lat="  +  coordinate[0]  +
"&longg=" + coordinate[1] + "&speed=" + coordinate[2] + " HTTP/1.1")
        sendAT("Host: www.arfpi.net")
        sendAT2("Connection: keep-alive")
        res = answer2()
        SER.send(res)
        task = 0
        if (res.find("NO CARR") > -1):
            stepbystep = 2
        if (res.find("ERROR") > -1):
            stepbystep = 0
        if (res.find("stopprog***") > -1):
            time.sleep(2)
            MDM.getDCD()
            MDM.send('+++',10)
            res = answer()
            loop = 0
        if (res.find("takephot***") > -1):
            time.sleep(2)
```

```
            MDM.getDCD()
            MDM.send('+++',10)
            res = answer()
            SER.send(res)
            stepbystep = 6
        time.sleep(3)

    if (stepbystep == 5):
        lastcommand = "AT#SGACT=1,0"
        res = ATCommand()
        if (res.find("ERROR") > -1):
            stepbystep = 0
        else:
            stepbystep = 2

    if (stepbystep == 205):
        lastcommand = "AT#SGACT=1,0"
        res = ATCommand()
        if (res.find("ERROR") > -1):
            stepbystep = 0
        else:
            stepbystep = 200

    if (stepbystep == 6):
        lastcommand = 'AT#FTPTO=400'
        res = ATCommand()
        if (res.find("ERROR") > -1):
            stepbystep = 200
        else:
            stepbystep = 65

    if (stepbystep == 65):
        lastcommand = 'AT#FTPOPEN="xx.xx.xx.xx","xxxxxx","xxxxx",1'
        res = ATCommand()
        if (res.find("ERROR") > -1):
            stepbystep = 0
        else:
            stepbystep = 7

    if (stepbystep == 7):
        lastcommand = 'AT#FTPTYPE=1'
        sendtime = 5
        res = ATCommand()
        if (res.find("ERROR") > -1):
            stepbystep = 200
        else:
            stepbystep = 8

    if (stepbystep == 8):
        SER.send("File is creating.." + CRLF)
        lastcommand = 'AT#FTPPUT="vtr.txt"'
        res = ATCommand()
        if (res.find("ERROR") > -1):
            stepbystep = 200
        else:
            stepbystep = 9

    if (stepbystep == 9):
        b = SER.read()
        while(b != ''):
```

```
            b = SER.read()

        FTPFILECONTENT= ''
        SER.sendbyte(0x36)
        SER.sendbyte(0x5F)
        SER.sendbyte(0x56)
        SER.sendbyte(0x5F)
        FTPFILECONTENT = SER.read()
        while(FTPFILECONTENT.find('FFD9') == -1):
            FTPFILECONTENT =  SER.read()
            a = MDM.send(FTPFILECONTENT, 0)
        stepbystep = 10
        task = 1

    if (stepbystep == 10):
        time.sleep(2)
        MDM.getDCD()
        MDM.send('+++',10)
        res = answer2()
        SER.send(res)
        lastcommand = 'AT#FTPCLOSE'
        res = ATCommand()
        if (res.find("ERROR") > -1):
            stepbystep = 0
        else:
            stepbystep = 3
```

Appendix 4: Abbreviations

API	: Application Programming Interface
GIS	: Geographic Information System
GPS	: Global Positioning System
GSM	: Global System for Mobile Communications
MVC	: Model View Controller
RWD	: Responsive Web Design
WMS	: Web Map Service
VTRS	: Vehicle Tracking and Routing System