# VEHICLE TRACKING USING GPS

A Project Report Submitted to

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY**

In partial fulfilment of the requirements for the award of the degree of

**BACHELOR OF TECHNOLOGY**

**IN**

**ELECTRONICS AND COMMUNICATION ENGINEERING**

SUBMITTED BY

| | |
|---|---|
| **D ARUN KUMAR** | **(15BD1A0417)** |
| **S S GAURAV** | **(15BD1A0448)** |
| **P.V.GAYATHRI** | **(15BD1A0440)** |

Under the Guidance of

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY
(Approved by AICTE, Affiliated to JNTUH)
Narayanaguda,Hyderabad
Academic Year: 2018-19

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

## CERTIFICATE

This is to certify that this bonafide record of the project report titled **"VEHICLE TRACKING USING GPS"** which is being presented as the Major Project report by

| | |
|---|---|
| **D ARUN KUMAR** | **(15BD1A0417)** |
| **P.V.GAYATHRI** | **(15BD1A0440)** |
| **S S GAURAV** | **(15BD1A0448)** |
| **S V S HARSHA** | **(15BD1A0450)** |

In partial fulfilment for the award of the degree of **Bachelor of Technology** in **Electronics and Communication Engineering** affiliated to the **Jawaharlal Nehru Technology University, Hyderabad.**

**Internal Guide** **Head of the Department**

Mr. K. Anil Kumar Dr. S.J.S. Antony M.B.A.,M.E.,Ph.D

**Submitted for the Project Viva Voce Examination held on …………….**

**Internal Examiner** **External Examiner**

# DECLARATION

We hereby declare that the results embodied in this dissertation entitled **"VEHICLE TRACKING USING GPS"** has been carried out by us together during the academic year 2018-2019 as a partial fulfilment of the award of the B. Tech degree in Electronics and Communication Engineering from JNTUH. We have not submitted this report to any other university or organization for award of any other degree.

| | |
|---|---|
| **D ARUN KUMAR** | **(15BD1A0417)** |
| **P.V.GAYATHRI** | **(15BD1A0440)** |
| **S S GAURAV** | **(15BD1A0448)** |
| **S V S HARSHA** | **(15BD1A0450)** |

# ACKNOWLEDGEMENT

The satisfaction and euphoria that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant encouragement and guidance have been a source of inspiration throughout the course of the project work. I am glad to have the opportunity of expressing our gratitude to all of them.

We render our thanks to **Dr. Maheshwar Dutta** B.E. M Tech, Ph.D, Principal who encouraged us to do the Project.
We are grateful to **Mr. Neil Gogte**, Director for facilitating all the amenities required for carrying out this project.
We express our sincere gratitude to **Mr. Nitin Kumar**, Director and **Mrs. Deepa Abhishek Ganu**, Director for providing an excellent environment in the college.

We are also thankful to **Dr. S.J.S Antony**, B.E, M.BA,M.E, Ph.D. and Head of Department for providing us with both time and amenities to make this project a success within the given schedule.

We are also thankful to **Mr. K. Anil Kumar**, Internal Guide for his valuable guidance an encouragement given to us throughout the project work.

We sincerely thank **Prof. Vinay Patankar, Mr.Prabhu Deshpande, Mr.Ramesh Mr.Madhukar, Mr.Shekar and Mrs. Rathna** of Research and Development, KMIT for their encouragement and support for us.

We would like to thank the entire ECE Department faculty who helped us directly and indirectly in the completion of the project.

| | |
|---|---|
| **D ARUN KUMAR** | **(15BD1A0417)** |
| **P.V.GAYATHRI** | **(15BD1A0440)** |
| **S S GAURAV** | **(15BD1A0448)** |
| **S V S HARSHA** | **(15BD1A0450)** |

# INDEX

# Abstract

An efficient vehicle tracking system is designed and implemented for tracking the movement of any equipped vehicle from any location at any time. The proposed system made good use of a popular technology that combines a Smartphone application with a microcontroller. This will be easy to make and inexpensive compared to others. The designed in-vehicle device works using Global Positioning System (GPS) and IOT technology that is one of the most common ways for vehicle tracking. The device is embedded inside a vehicle whose position is to be determined and tracked in real-time. A microcontroller is used to control the GPS and WiFi modules. The vehicle tracking system uses the GPS module to get geographic coordinates at regular time intervals. The Wifi module is used to transmit and update the vehicle location to a database. A Smartphone application is also developed for continuously monitoring the vehicle location. The Google Maps API is used to display the vehicle on the map in the Web application. Thus, users will be able to continuously monitor a moving vehicle on demand using the Smartphone application and determine the estimated distance and time for the vehicle to arrive at a given destination. In order to show the feasibility and effectiveness of the system, this paper presents experimental results of the vehicle tracking system and some experiences on practical implementations.

# CHAPTER 01
# INTRODUCTION

## 1.1 INTRODUCTION:

The main aim of Vehicle Tracking System is to give security to the vehicle and driver. The GPS are highly useful now a days, this system enables the owner to observe and track his vehicle and find out vehicle movement and its past activities of vehicle. The hardware is fitted on to the vehicle in such a manner that it is not visible to anyone who is inside or outside of the vehicle. When the vehicle is stolen the vehicle location data can be tracked to find the location and can be informed to police for further action. Vehicle Tracking System carries the input from GPS device and sends the obtained data to the cloud using Wi-Fi module and further to desired device using IOT. .Generally this system is meant to be installed for the four wheelers but for country like India where majority of the people using two wheelers, here is the cheapest source of an anti-theft tracking system. Vehicle tracking systems are commonly used by fleet operators for fleet management functions such as routing, dispatch, on-board information and security. Other applications include monitoring driving behavior, such as an employer of an employee, or a parent with a teen driver.

## 1.2 Proposed System:

The proposed system is used for positioning and navigating the vehicle with an accuracy of 10m. The Exact location is indicated in the form of latitude and longitude along with the exact Navigated track on Google map. The system tracks the location of particular vehicle and sends to users mobile in form of data and also to microcontroller. The arrived data, in the form of latitude and longitude is used to locate the Vehicle on the Google maps using web application.

## 1.3 VEHICLE TRACKING FEATURES:

It is mainly benefit for the companies which are based on transport system. Since it can show the position of all vehicles in real time, so that they can create the expected data accordingly. These tracking system can store the whole data where the vehicle had gone, where did it stop, how much time it take at every stop and can create whole data analysis.

It is also used in buses and trains, to estimate how far are they, how much time it takes for them to come to a particular stop. These systems are used to data capture, data storage, data analysis and finally data transfer. By adding additional sensors such as temperature sensor and infrared sensors the system can be enabled to detect fire , theft and obstacles.

# CHAPTER 02

# System Overview

## 2.1. History and Literature Review:

Global Positioning Systems (GPS) were designed by the United States Government and military, which the design was intended to be used as surveillance. The GPS was invented as a collaborative effort by the United States‟ Department of Defense and Dr. Ivan Getting as a means to create a satellite course-plotting system, primarily used for navigation purposes .At that time, the GPS project cost approximately $12 billion for the design and launch of 18 satellites, six in each of the orbital planes spaced 120 degrees apart, and their ground stations. GPS uses these satellites as reference points to determine and give the accurate geographical positions on map .The idea for a global positioning system was initially planned to be used by military and intelligence organizational during the Cold War, with the introduction of the project stemming from the Soviet-launched spacecraft Sputnik. Since its introduction in the 1960s, GPS has developed into a larger and more advanced satellite network constellation that orbits Earth at fixed points in space to send signals to anyone with a GPS receiver. The signals carry a time code and geographic data point that enables us to display a device’s exact position anywhere on the planet .The design of GPS is partly similar to the design of-ground-based radio navigation systems, such as LORAN and the Decca Navigator, developed in the early 1940s and were used during World War II. Additional inspiration for the GPS system came when the Soviet Union launched the first Sputnik in 1957. A team of U.S. scientists led by Dr. Richard B. Kershner were monitoring Sputnik's radio transmissions.

*Fig: Dr. Richard B. Kershner*

They discovered that, because of the Doppler Effect, the frequency of the signal being transmitted by Sputnik was higher as the satellite approached and lower as it moves away from them. They realized that since they knew their exact location on the globe, by measuring the Doppler distortion it was possible top in point where the satellite was along its orbit. The first satellite navigation system was first successfully tested in 1960. It delivers a navigational fix approximately once per hour using a constellation of five satellites. In 1967, the U.S. Navy introduced the timation satellite which demonstrated the ability to place accurate clocks in space that is the technology used by the GPS system. In the 1970s, the ground-based Omega Navigation System, based on signal phase comparison, became the first world-wide radio navigation system. In February 1978the first experimental Block-I GPS satellite was launched. The GPS satellites were initially manufactured by Rockwell International and are now mass-produced by Lockheed Martin. In 1983, after Soviet interceptor aircraft shot down the civilian airliner KAL 007 in restricted Soviet airspace, killing all 269 people on board, U.S. President Ronald Reagan announced that the GPS system would be made available for civilian uses once it was completed. Hence, the government signed a treaty to allow civilians to buy GPS units also only the civilians would get precise downgraded ratings. The oldest GPS satellite still in operation was launched in August 1991. By December 1993 the GPS system achieved initial operational capability and a complete constellation of 24 satellites was in orbit by January 17, 1994 .In the initial period of tracking only two radios were used to exchange the information. One radio was attached to the vehicle while another at base station by which drivers were enabled to talk to their masters. Fleet operator could identify the progress through their routes. The early technology also has some limitation. It was restricted by the distance which became a hurdle in accuracy and better connectivity between driver and fleet operators. Base station was dependent on the driver for the information and a huge size fleet could not have been managed depending on man-power only. The scene of vehicle tracking underwent a change with the arrival of GPS technology. This reduced the dependence on man-power. Most of the work of tracking became electronic. Computers proved a great help in managing a large fleet of vehicle. This also made the information authentic. As this technology was available at affordable cost all whether small or big fleet could take benefit of this technology. Because of the accessibility of the device computer tracking facilities has come to stay and associated with enhanced management. Today each vehicle carries tracking unit which is monitored from the base station. Base station receives the data from the unit. All these facilities require a heavy investment of capital for the installation of the infrastructure of tracking system for monitoring and dispatching.

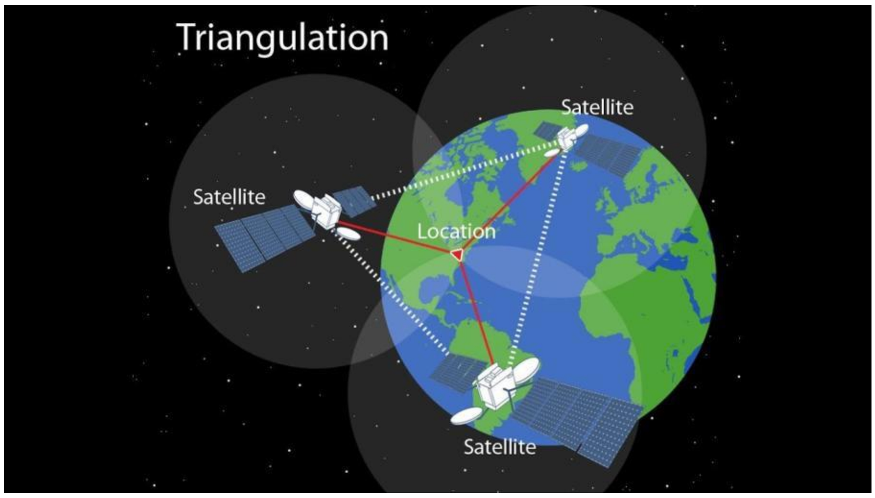


*Fig: Triangulation Concept of working of GPS*

Today's GPS applications have vastly developed. It is possible to use the Global Positioning Systems to design expense reports, create time sheets, or reduce the costs of fuel consumption. We can also use the tracking devices to increase efficiency of employee driving. The GPS unit allows us to create Geo-Fences about a designated location, which gives us alerts once the driver passes through that location. This means we have added security combined with more powerful customer support for our workers. Nowadays GPS units are great tracking devices that help fleet managers stay in control of their business. The applications in today's GPS units make it possible to take full control of any company. It is clear that the tracking devices offer many benefits to companies, since we can build automated expense reports anytime. GPS units do more than just allow companies to create reports. These devices also help to put an end to thieves. According to recent reports, crime is at a high, which means that car theft is increasing. If we have the right GPS unit, we can put an end to car thefts because we can lock and unlock our car anytime we want to. GPS is small tracking device that is installed in a car and it will supply feedback data from tracking software that loads from a satellite. In this paper GPS based vehicle navigation system is implemented. This is done by fetching the information of the vehicle like location, distance, etc. by using GPS and GSM or IOT. The information of the vehicle is obtained after every specified time interval defined by the user. Then this periodic information of location is transmitted to monitoring or tracking server. This transmitted information is displayed on the display unit by using the Google earth to display the vehicle location in the electronic Google maps.

This system uses Global Positioning System (GPS) which is used to receive the coordinates of latitude and longitude form the satellite during the critical information. We all know that tracking system is now-a-days a very important in modern world. This system can be used in the monitoring our car, also in tracking the theft of the vehicle and in many more other applications. This system uses microcontroller, Global Positioning System (GPS) and Internet of Things. Only one GPS device is used in this system and Wi-Fi module enable a two way communication process. From the above mentioned vehicle tracking techniques we can say that each technique is appropriate with its function but in some system we need continuous net access and this system can go down if net fails. In the first system the GPS tracks the vehicle location and send it to the controller and the Google maps display the location of the Vehicle on the display unit, this system is useless without net because the location of vehicle can only be presented by the Google maps. By considering all these factors the upcoming implementation should introduce many more facilities which will make the system user friendly and efficient.

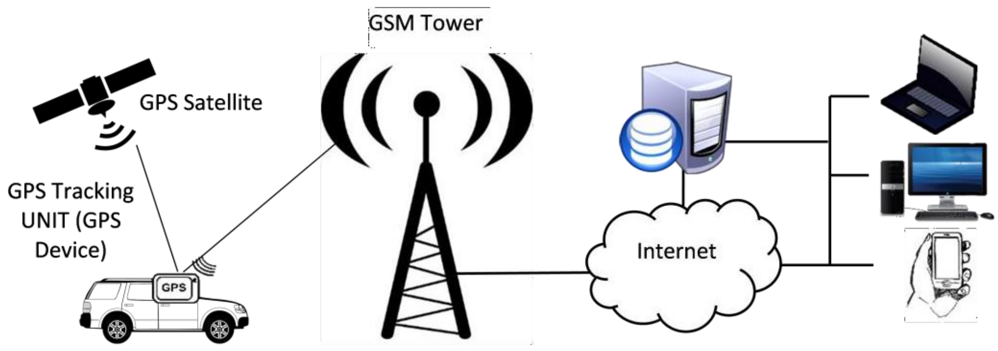


*Fig:Concept of vehicle tracking using GSM*

## 2.2 Different Technologies used in Tracking System

### 2.2.1 Active and passive tracking:

Several types of vehicle tracking devices exist. Typically they are classified as "passive" and "active". "Passive" devices store GPS location, speed, heading and sometimes a trigger event such as key on or off, door open or closed. Once the vehicle returns to a predetermined point, the device is removed and the data downloaded to a computer for evaluation. Passive systems include auto download type that transfer data via wireless download. "Active" devices also collect the same information but usually transmit the data in real-time via cellular or satellite networks to a computer or data center for evaluation. Passive trackers do not monitor movement in real-time. When using a passive GPS tracker, you will not be able to follow every last move that a tracked person or object makes. Instead, information that is stored inside of a passive tracker must be downloaded to a computer. Once tracking details have been downloaded, it is then possible to view tracking details. After we have gathered all of the information we need from a passive tracker, we can place the tracker back on the same (or different) vehicle. Aside from the fact that a passive tracking device is entirely reliable, the main reason people choose passive trackers is that these devices are less expensive than active trackers. Most passive GPS tracking devices are not attached to a monthly fee, which makes these trackers affordable. In contrast to passive devices, active GPS trackers will allow one to view tracking data in real-time. As soon as we place an active tracker on a vehicle, we will be able to view location, stop duration, speed, and other tracking details from the comfort of your home or office. Active GPS trackers are ideal when it comes to monitoring vehicle that need to be tracked at regular time interval. While active tracking devices are more expensive than passive devices (most come with monthly fees), this expensive is usually justified. An active GPS tracker that comes with a reliable interface (and excellent tracking software), and you will be able to track anything or anyone quickly and efficiently. When most people picture a GPS tracking device, they are picture a real-time tracker. These trackers can be attached to any object while a person monitors all activity from a home computer. For example, if you were to place a real-time tracker on a vehicle, you could then watch as the vehicle makes stops, takes alternate routes, and sits idling–all in real-time. GPS trackers that work on a real-time basis are usually considered "active" trackers, while those that do not include real-time tracking are considered "passive" trackers. There are many advantages associated with a real time tracker. The most important advantage is that the GPS locater is convenience. Rather than waiting to download data to a computer (as is the case with most passive trackers), a tracker that works in real-time does not require any waiting. Since real-time trackers come with software that allows a user to track an object in real-time, watching any object's progress is simply a matter of sitting at a computer.

Many modern vehicle tracking devices combine both active and passive tracking abilities: when a cellular network is available and a tracking device is connected it transmits data to a server; when a network is not available the device stores data in internal memory and will transmit stored data to the server later when the network becomes available again. Historically vehicle tracking has been accomplished by installing a box into the vehicle, either self-powered with a battery or wired into the vehicle's power system. For detailed vehicle locating and tracking this is still the predominant method; however, many companies are increasingly interested in the emerging cell phone technologies that provide tracking of multiple entities.

**2.2.2 Different Types of Tracking System:**

There are three main types of GPS vehicle tracking that are widely used. They all use active devices. They are:

1. Automatic Vehicle Location (AVL) system

2. Assisted Global Positioning System (AGPS)
3. Radio Frequency Identification (RFID)

- Automatic Vehicle Location (AVL) system-AVL system is an advanced method to track and monitor any remote vehicle with the device that receives and sends signals through GPS satellites. AVL comprises of Global Positioning System (GPS) and Geographic Information System (GIS) in order to provide the real geographic location of the vehicle. AVL system consists of PC-based tracking software to dispatch, a radio system, GPS receiver on the vehicle and GPS satellites. Among the two types of AVL, GPS-based and Signpost-based, GPS-based system is widely used. The tracking method uses GPS satellite to locate the vehicle equipped with GPS modem by sending satellite signals. The accuracy of the tracking method depends on the AVL system which provides the vehicle location with the accuracy of about 5m to 10m. The information transmitted by the tracking system to the base station is location, speed, direction, mileage, start and stop information and status of vehicle. The information of the vehicle is often transmitted to the central control system (base station) from the vehicle after every 60 seconds. If the base station receives the data, it displays it on a computerized map.GPS receiver on the vehicle receives the signals of its geographic location. The system also has some limitation; using the AVL system we cannot get accurate, complete and sufficient satellite data in dense urban areas or indoors and when transmission is blocked by natural obstructions (heavy tree cover) or many buildings. It can also occur in RF-shadowed environments and under unfriendly Radio Frequency (RF) conditions. Sometimes, a position fix can be impossible.

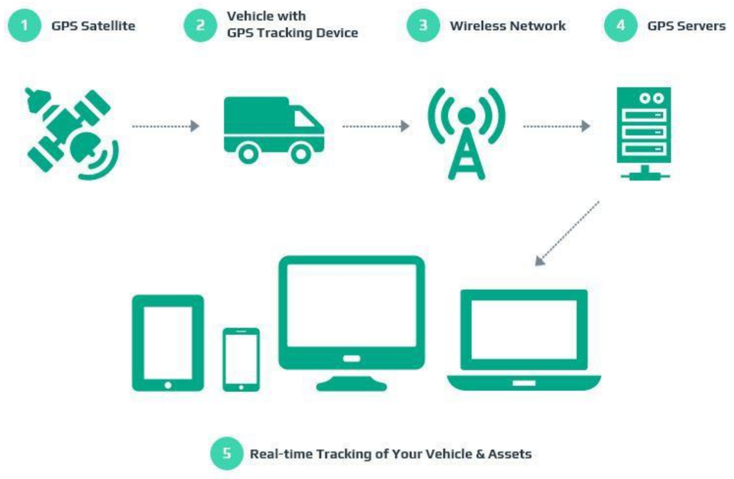

1 GPS Satellite
2 Vehicle with GPS Tracking Device
3 Wireless Network
4 GPS Servers
5 Real-time Tracking of Your Vehicle & Assets

*Fig : Concept diagram for Vehicle Tracking*

- Assisted GPS (AGPS) system-In AGPS system, a terrestrial RF network is used to improve the performance of GPS receivers as it provides information about the satellite constellation directly to the GPS receivers. AGPS uses both mobiles and cellular networks to locate the accurate positioning information. AGPS is used to overcome some limitations of GPS. With unassisted GPS, locating the satellites, receiving the data and confirming the exact position may take several minutes. The tracking method of AGPS uses GPS satellites to track the vehicles. A GPS receiver in vehicle is always in contact with 4 satellites (3 satellites determine latitude, longitude and elevation and the fourth provides element of time) hence it never fails to detect the location of a vehicle. Location of the vehicle is provided with accuracy of between 3m and 8m, and speed of 1km using this method. Information like Vehicle location, average speed, direction, path traversed in a selected period and alerts (Engaged/Unengaged, speed limit, vehicle breakdown and traffic jam) are delivered by the tracking system to the base station. The system provides continuous updates after every 10 seconds while the vehicle is in motion. It also provides data storage for up to 1 year. The location is retrieved from the GPS device and relayed as a SMS using the cell phone by the Client Node to the Base station. This system is more expensive than the AVL system as it gives continuous update of the vehicle location. If the user needs update after every 10 seconds then the subscription for this system is $1.33 per day per vehicle and if the user needs update after every 5 seconds it is $1.67 per day per vehicle. The system can provide further services like atomic time .There is a "panic" button. When pressed, you can contact an operator and he or she will help you out or keep you safe from accidents or hijacks. The system has also some limitations as GSM network is used to transmit data from the vehicle to the base station, the cost of sending SMS is a major concern to be considered

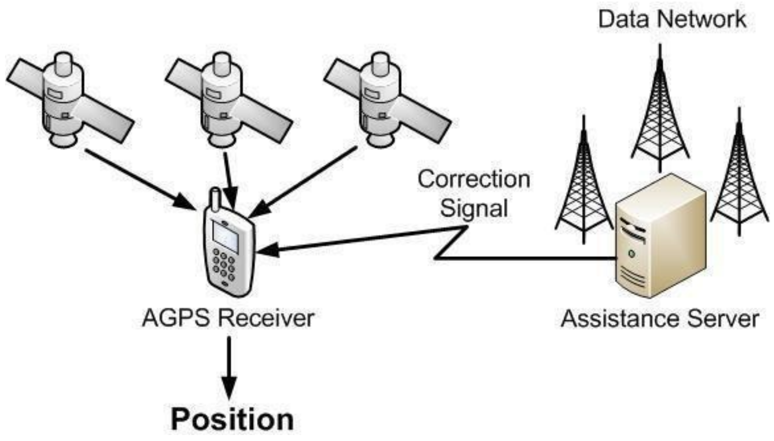


*Fig: Assisted GPS (AGPS) system*

- Radio Frequency Identification (RFID) System-RFID is an automatic identification method using devices called tags to store and remotely retrieves data. RFID uses radio waves to capture data from tags. The tracking method of RFID is comprised of three components: tag (passive, semi passive and active), reader (antenna or integrator) and software (middleware). RFID tag which contains microelectronic circuits sends the vehicle information to a remote RFID reader which is then read via the software. This system provides the location of the vehicle with the accuracy of 4m to 6m. Information such as location of the vehicle, mileage and speed are delivered by the tracking system to the centre. The information is updated every one minute. The information is sent to and received from RFID tags by a reader using radio waves. RFID reader, basically a radio frequency (RF) transmitter and receiver, is controlled by a microprocessor or digital signal processor (DSP). RFID reader with an attached antenna reads data from RFID tags.

*Fig:Tracking using RFID*

# Chapter 03

# Hardware Components

## 3.1 NodeMCU:

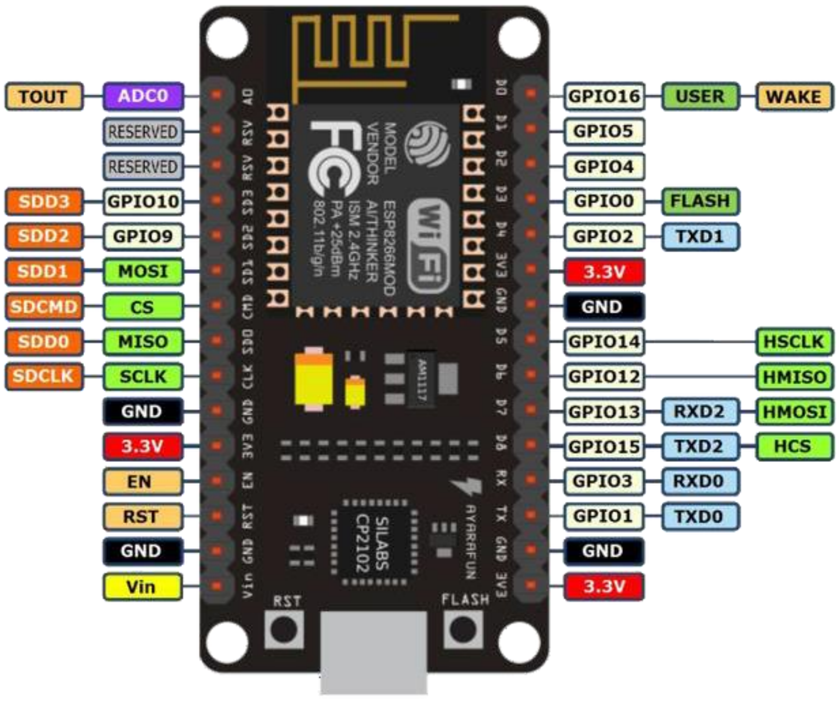


**FIG: Architecture of NodeMCU**

NodeMCU is an open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module. The term "NodeMCU" by default refers to the firmware rather than the development kits. The firmware uses the Lua scripting language. It is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson and SPIFFS.

NodeMCU was created shortly after the ESP8266 came out. On December 30, 2013, Espressif Systems began production of the ESP8266. The ESP8266 is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core, widely used in IoT applications NodeMCU started on 13 Oct 2014, when Hong committed the first file of nodemcu-firmware to GitHub.] Two months later, the project expanded to include an open-hardware platform when developer Huang R committed the gerber file of an ESP8266 board, named devkit v0.9. Later that month, Tuan PM ported MQTT client library from Contiki to the ESP8266 SoC platform, and committed to NodeMCU project, then NodeMCU was able to support the MQTT IoT protocol, using Lua to access the MQTT broker. Another important update was made on 30 Jan 2015, when Devsaurus ported the u8glib to NodeMCU project, enabling NodeMCU to easily drive LCD, Screen, OLED, even VGA displays.

In summer 2015 the creators abandoned the firmware project and a group of independent contributors took over. By summer 2016 the NodeMCU included more than 40 different modules. Due to resource constraints users need to select the modules relevant for their project and build a firmware tailored to their needs.

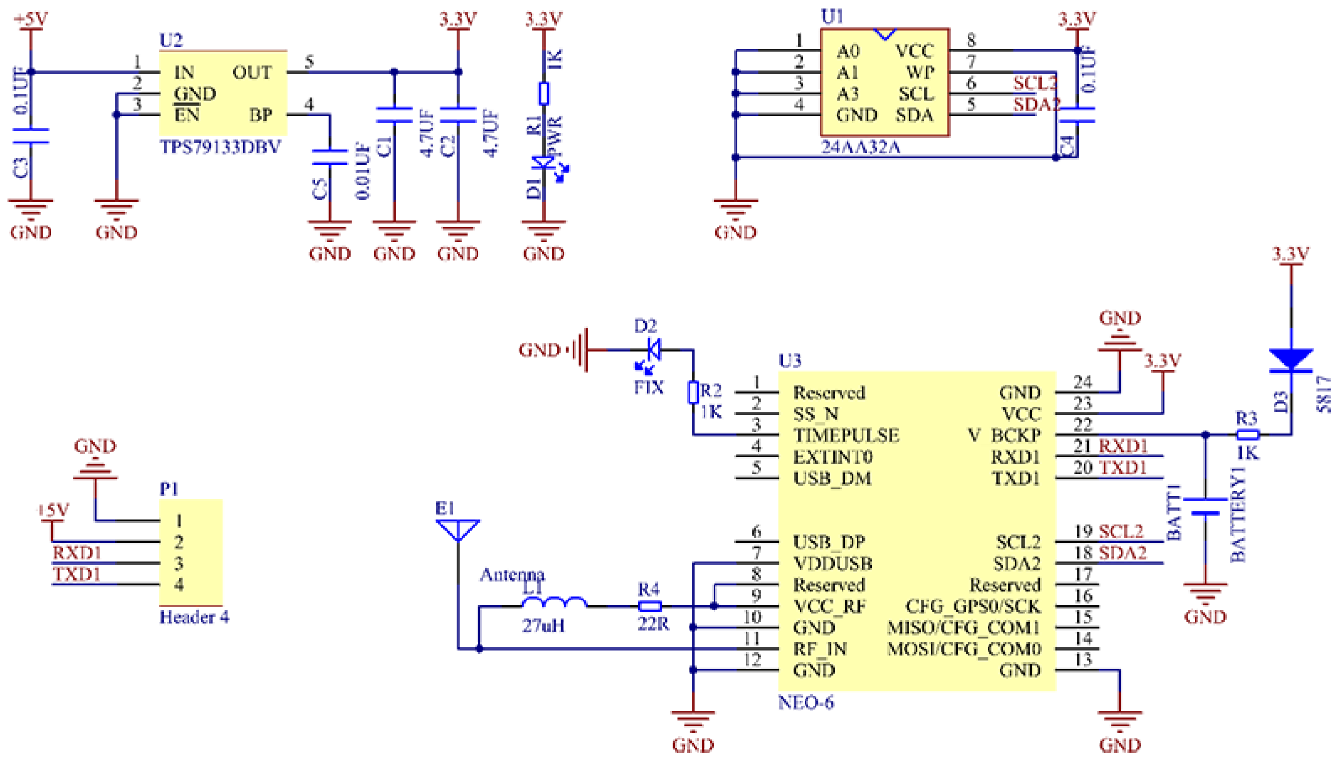
**ESP8266 Arduino Core**

As Arduino.cc began developing new MCU boards based on non-AVR processors like the ARM/SAM MCU and used in the Arduino Due, they needed to modify the Arduino IDE so that it would be relatively easy to change the IDE to support alternate tool chains to allow Arduino C/C++ to be compiled for these new processors. They did this with the introduction of the Board Manager and the SAM Core. A "core" is the collection of software components required by the Board Manager and the Arduino IDE to compile an Arduino C/C++ source file for the target MCU's machine language. Some ESP8266 enthusiasts developed an Arduino core for the ESP8266 WiFi SoC, popularly called the "ESP8266 Core for the Arduino IDE". This has become a leading software development platform for the various ESP8266-based modules and development boards, including NodeMCUs.

## 3.2 GPS Module (Neo 6m):

**Fig : Neo 6m**

The NEO-6M GPS module is a well-performing complete GPS receiver with a built-in 25 x 25 x 4mm ceramic antenna, which provides a strong satellite search capability. With the power and signal indicators, you can monitor the status of the module. Thanks to the data backup battery, the module can save the data when the main power is shut down accidentally. Its 3mm mounting holes can ensure easy assembly on your aircraft, which thus can fly steadily at a fixed position, return to Home automatically, and automatic waypoint flying, etc. Or you can apply it on your smart robot car for automatic returning or heading to a certain destination, making it a real "smart" bot! The schematic diagram of the module is shown as below.



**Features**

1) A complete GPS module with an active antenna integrated, and a built-in EEPROM to save configuration parameter data.

2) Built-in 25 x 25 x 4mm ceramic active antenna provides strong satellite search capability.

3) Equipped with power and signal indicator lights and data backup battery.

4) Power supply: 3-5V; Default baud rate: 9600bps.

5) Interface: RS232 TTL

**Test**

In this test, we will send the positioning data collected by the NEO-6M GPS Module to the software on the PC, and compare this result with that of a standard GPS device. Thus we can know whether this module works or not.

**Preparations**

1 x USB-to-Serial Module (e.g. FTDI Module)
1 x USB Cable
Laptop

1 x NEO-6M GPS Module

**Note:** Pay attention to the wiring between the power and ground of the testing board, like VCC and GND, to prevent a short circuit.

**Procedures**

1) Connect the NEO-6M GPS Module and the FTDI Module as shown below.

| NEO-6M GPS Module | FTDI Module |
|---|---|
| TX | RX |
| RX | TX |
| VCC | VCC |
| GND | GND |

2) Then connect the FTDI Module to the computer with a USB cable. Observe and you'll see the red LED blinking, then take the GPS Module outdoor (move them all to a position where the GPS module can receive the signal from satellites).

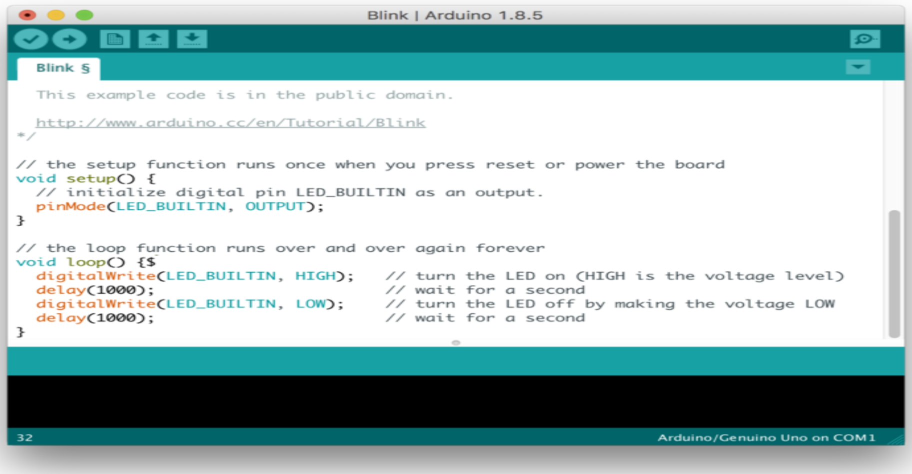
# Chapter 04

# Software Components

## 4.1 Arduino IDE:

The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java. It is used to write and upload programs to Arduino compatible boards, but also, with the help of 3rd party cores, other vendor development boards.

The source code for the IDE is released under the GNU General Public License, version 2. The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware



```
  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/Blink
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {$
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

.

Fig: Screenshot of Arduino IDE showing Blink program

## 4.2 ThingSpeak:

According to its developers, "ThingSpeak is an open-source Internet of Things (IoT) application and API to store and retrieve data from things using the HTTP protocol over the Internet or via a Local Area Network. ThingSpeak enables the creation of sensor logging applications, location tracking applications, and a social network of things with status updates".

ThingSpeak was originally launched by IO Bridge in 2010 as a service in support of IoT applications.

ThingSpeak has integrated support from the numerical computing software MATLAB from Math Works, allowing ThingSpeak users to analyze and visualize uploaded data using MATLABwithout requiring the purchase of a MATLAB license from Mathworks.

ThingSpeak has a close relationship with Mathworks, Inc. In fact, all of the ThingSpeak documentation is incorporated into the Mathworks' Matlab documentation site and even enabling registered Mathworks user accounts as valid login credentials on the ThingSpeak website. The terms of service and privacy policy of ThingSpeak.com are between the agreeing user and Mathworks, Inc.

ThingSpeak has been the subject of articles in specialized "Maker" websites like Instructables, Codeproject, and Channel 9.

## 4.3. Google Maps:

**Google Maps** is a web mapping service developed by Google. It offers satellite imagery, aerial photography, street maps, 360° panoramic views of streets (Street View), real-time traffic conditions, and route planning for traveling by foot, car, bicycle and air (in beta), or public transportation.

Google Maps began as a C++ desktop program at Where 2 Technologies. In October 2004, the company was acquired by Google, which converted it into a web application. After additional acquisitions of a geospatial data visualization company and a real time traffic analyzer, Google Maps was launched in February 2005. The service's front end utilizes JavaScript, XML, and Ajax. Google Maps offers an API that allows maps to be embedded on third-party websites, and offers a locator for businesses and other organizations in numerous countries around the world. Google Map Maker allowed users to collaboratively expand and update the service's mapping worldwide but was discontinued from March 2017. However, crowd sourced contributions to Google Maps were not discontinued as the company announced those features will be transferred to the Google Local Guides program.

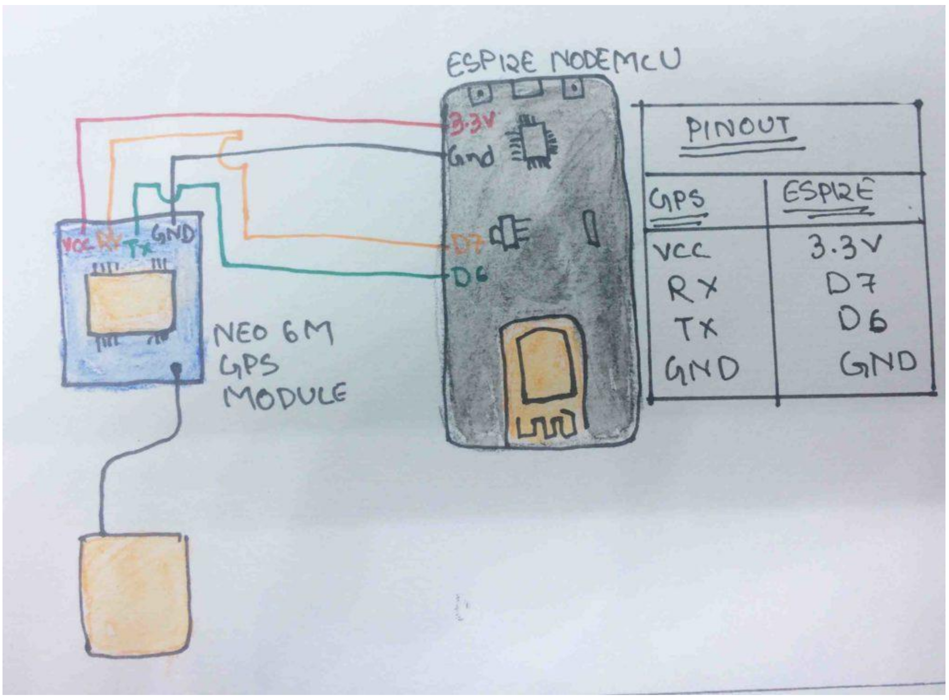
Google Maps' satellite view is a "top-down" or "birds eye" view; most of the high-resolution imagery of cities is aerial photography taken from aircraft flying at 800 to 1,500 feet (240 to 460 m), while most other imagery is from satellites. Much of the available satellite imagery is no more than three years old and is updated on a regular basis. Google Maps used a variant of the Mercator projection, and therefore cannot accurately show areas around the poles. However, in August 2018, the desktop version of Google Maps was updated to show a 3D globe.

Google Maps for Android and IOS devices was released in September 2008 and features GPS turn-by-turn navigation along with dedicated parking assistance features. In August 2013, it was determined to be the world's most popular app for smartphones, with over 54% of global smartphone owners using it at least once.

# Chapter 05

# Working

## Working Of Vehicle Tracker:



PINOUT

| GPS | ESP12E |
|---|---|
| VCC | 3.3V |
| RX | D7 |
| TX | D6 |
| GND | GND |

***Fig : circuit diagram of vehicle tracker***

This vehicle tracking system takes input from GPS and sends it through the NodeMCU to ThingSpeak and further data can extracted using web application in mobile / laptop. Vehicle Tracking System is one of the biggest technological advancements to track the activities of the vehicle. The security system uses Global Positioning System GPS, to find the location of the monitored or tracked vehicle and then uses satellite or radio systems to send to send the coordinates and the location data to the monitoring center. At monitoring center various software's are used to plot the Vehicle on a map. In this way the Vehicle owners are able to track their vehicle on a real-time basis. Due to real-time tracking facility, vehicle tracking systems are becoming increasingly popular among owners of expensive vehicles.

# Chapter 06

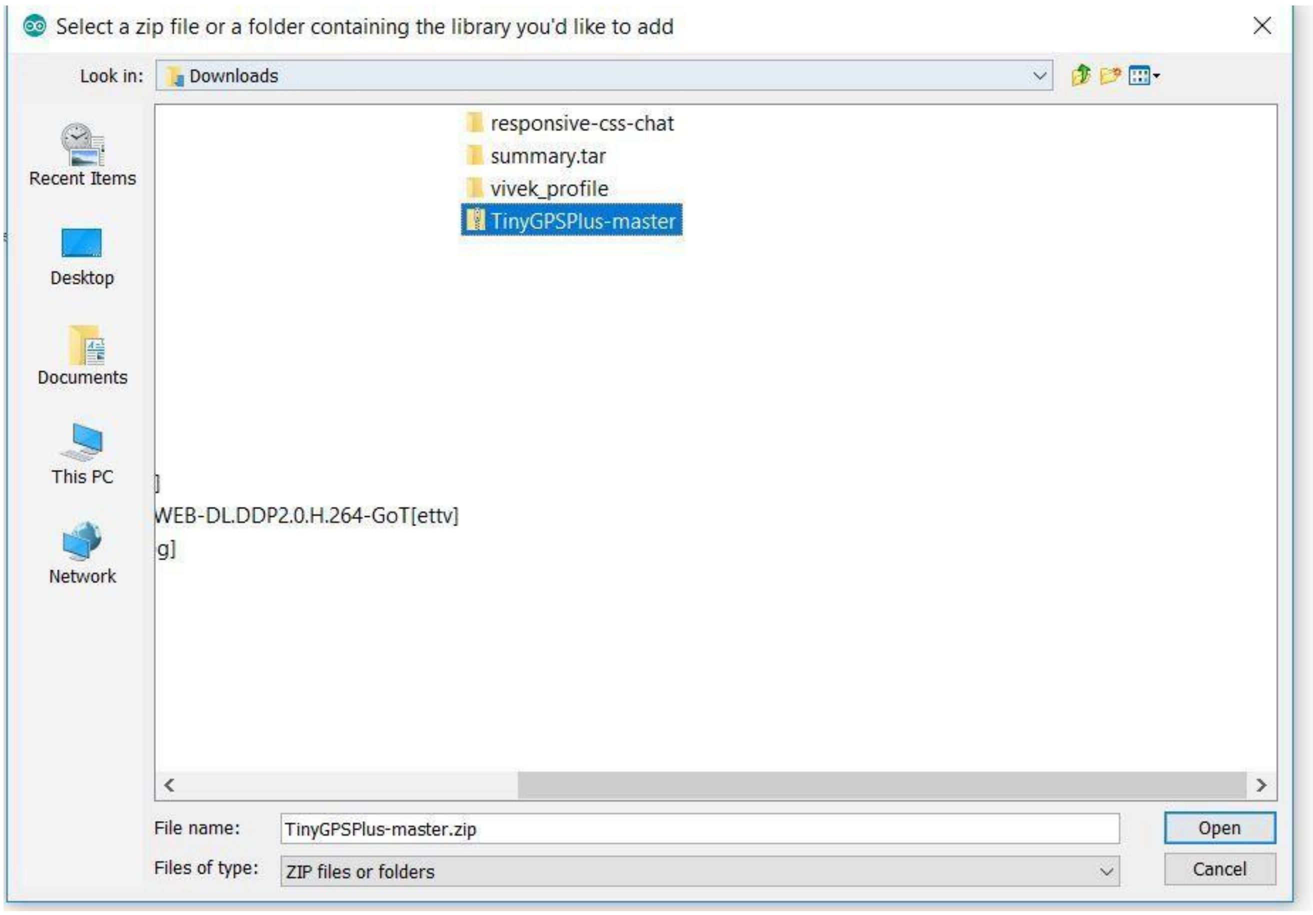
# Code

## 6.1 Programming ESP12E NodeMCU Module to upload GPS data on Thingspeak

1. You will need the "Tiny GPS Plus" library for this program. Visit the following link and add it to the arduino library.

https://github.com/mikalhart/TinyGPSPlus

Download the zip and include it in the arduino library.



File name: TinyGPSPlus-master.zip

Files of type: ZIP files or folders

## Program to read gps data and upload to thingspeak

```
#include <TinyGPS++.h>

#include <SoftwareSerial.h>

#include "ThingSpeak.h"

#include <ESP8266WiFi.h>


/*

  This sample sketch demonstrates the normal use of a TinyGPS++ (TinyGPSPlus) object.

  It requires the use of SoftwareSerial, and assumes that you have a

  4800-baud serial GPS device hooked up on pins 7(rx) and 6(tx).

*/

static const int RXPin = D7, TXPin = D6;

static const uint32_t GPSBaud = 9600;


// repace your wifi username and password

const char* ssid     = "your SSID";

const char* password = "your password";


unsigned long myChannelNumber = THINGSPEAK CHANNEL NUMBER;

const char * myWriteAPIKey = "THINGSPEAK WRITE API KEY";


// The TinyGPS++ object

TinyGPSPlus gps;

WiFiClient  client;


// The serial connection to the GPS device
```

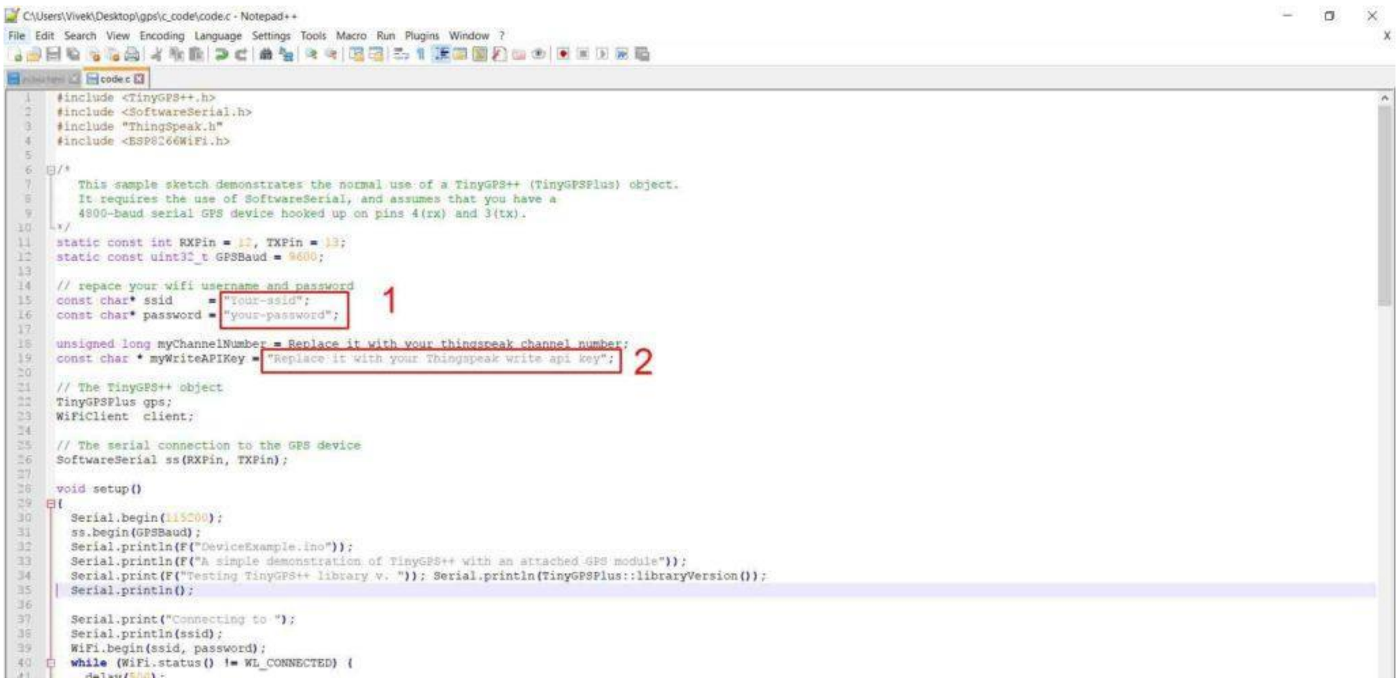
```
SoftwareSerial ss(RXPin, TXPin);
 void setup()
{
  Serial.begin(115200);
  ss.begin(GPSBaud);
  Serial.println(F("DeviceExample.ino"));
  Serial.println(F("A simple demonstration of TinyGPS++ with an attached GPS module"));
  Serial.print(F("Testing TinyGPS++ library v. ")); Serial.println(TinyGPSPlus::libraryVersion());
  Serial.println();
 /*CODE TO CONNECT NODE MCU TO YOUR DEFINED SSID*/
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
   Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  Serial.print("Netmask: ");
  Serial.println(WiFi.subnetMask());
  Serial.print("Gateway: ");
  Serial.println(WiFi.gatewayIP());
  ThingSpeak.begin(client);
```

```
}


void loop()

{

  /* This sketch displays information every time a new sentence is correctly encoded.*/

  while (ss.available() > 0)

    if (gps.encode(ss.read()))

      displayInfo();


  if (millis() > 5000 && gps.charsProcessed() < 10)

  {

    Serial.println(F("No GPS detected: check wiring."));

    while(true);

  }

}

/*METHOD TO DISPLAY LATITUDE AND LONGITUDE ON SERIAL MONTOR AND WRITE
THEM TO THINGSPEAK FIELDS.*/

void displayInfo()

{

 // Serial.print(F("Location: "));

  if (gps.location.isValid())

  {

    double latitude = (gps.location.lat());

    double longitude = (gps.location.lng());

    String latbuf;
    latbuf += (String(latitude, 6));
    Serial.println(latbuf);
    String lonbuf;
    lonbuf += (String(longitude, 6));
    Serial.println(lonbuf);
    ThingSpeak.setField(1, latbuf);
    ThingSpeak.setField(2, lonbuf);
    ThingSpeak.writeFields(myChannelNumber, myWriteAPIKey);
    delay(20000);
      }
  else
  {
    Serial.print(F("INVALID"));
  }
```

**/*CODE TO GET DATE AND TIME*/**

```
 Serial.print(F("  Date/Time: "));
 if (gps.date.isValid())
 {
  Serial.print(gps.date.month());
  Serial.print(F("/"));
  Serial.print(gps.date.day());
  Serial.print(F("/"));
  Serial.print(gps.date.year());
 }
 else
 {
    Serial.print(F("INVALID"));
  }
  Serial.print(F(" "));
  if (gps.time.isValid())
  {
    if (gps.time.hour() < 10) Serial.print(F("0"));
    Serial.print(gps.time.hour());
    Serial.print(F(":"));
    if (gps.time.minute() < 10) Serial.print(F("0"));
    Serial.print(gps.time.minute());
    Serial.print(F(":"));
    if (gps.time.second() < 10) Serial.print(F("0"));
    Serial.print(gps.time.second());
    Serial.print(F("."));
    if (gps.time.centisecond() < 10) Serial.print(F("0"));
    Serial.print(gps.time.centisecond());
  }
  else
  {
    Serial.print(F("INVALID"));
  }
  Serial.println();
}
```

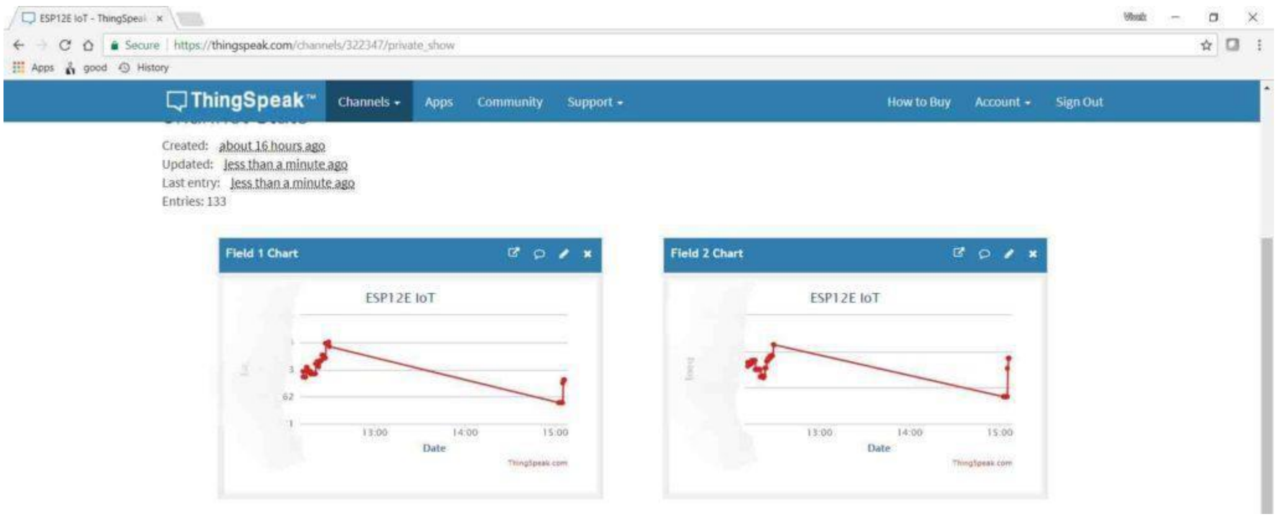
3. Replace the following things with your own thingspeak credentials:


C:\Users\Vivek\Desktop\gps\c_code\code.c - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
code.c
1
2

```
#include <TinyGPS++.h>
#include <SoftwareSerial.h>
#include "ThingSpeak.h"
#include <ESP8266WiFi.h>

/*
   This sample sketch demonstrates the normal use of a TinyGPS++ (TinyGPSPlus) object.
   It requires the use of SoftwareSerial, and assumes that you have a
   4800-baud serial GPS device hooked up on pins 4(rx) and 3(tx).
*/
static const int RXPin = 12, TXPin = 13;
static const uint32_t GPSBaud = 9600;

// repace your wifi username and password
const char* ssid     = "Your-ssid";
const char* password = "your-password";

unsigned long myChannelNumber = Replace it with your thingspeak channel number;
const char * myWriteAPIKey = "Replace it with your Thingspeak write api key";

// The TinyGPS++ object
TinyGPSPlus gps;
WiFiClient  client;

// The serial connection to the GPS device
SoftwareSerial ss(RXPin, TXPin);

void setup()
{
  Serial.begin(115200);
  ss.begin(GPSBaud);
  Serial.println(F("DeviceExample.ino"));
  Serial.println(F("A simple demonstration of TinyGPS++ with an attached GPS module"));
  Serial.print(F("Testing TinyGPS++ library v. ")); Serial.println(TinyGPSPlus::libraryVersion());
  Serial.println();

  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
```

4. Save the program and flash ESP12E NodeMCU Module with this program.

Now, wait till your Neo 6M GPS modules led has started blinking. It might take around 2-10 minutes but it will start blinking. Have patience. And then go check your thingspeak channel. If you have done everything right, then ESP12E NodeMCU will start updating the channel with a delay of 15 seconds. And you will get something like show in the below image:

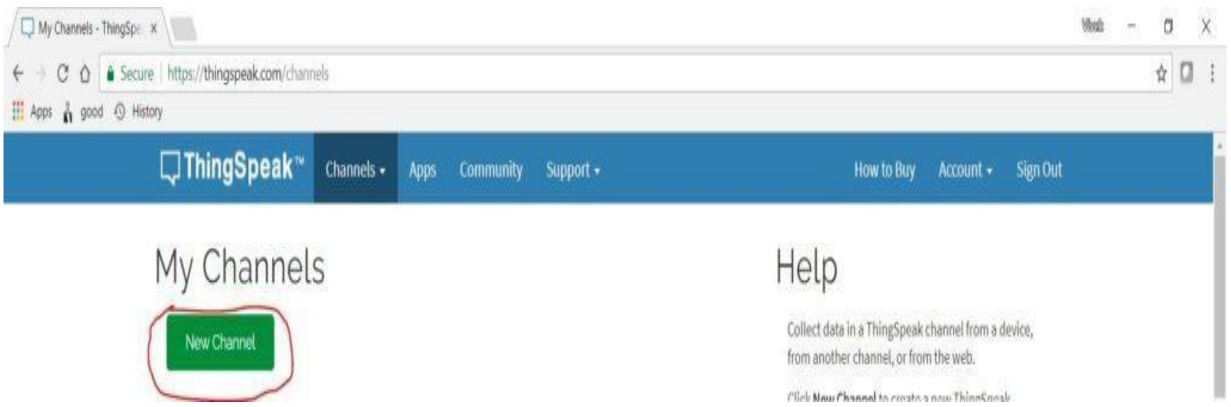


Created: about 16 hours ago
Updated: less than a minute ago
Last entry: less than a minute ago
Entries: 133

If you don’t have THINGSPEAK channel follow below steps to create one.

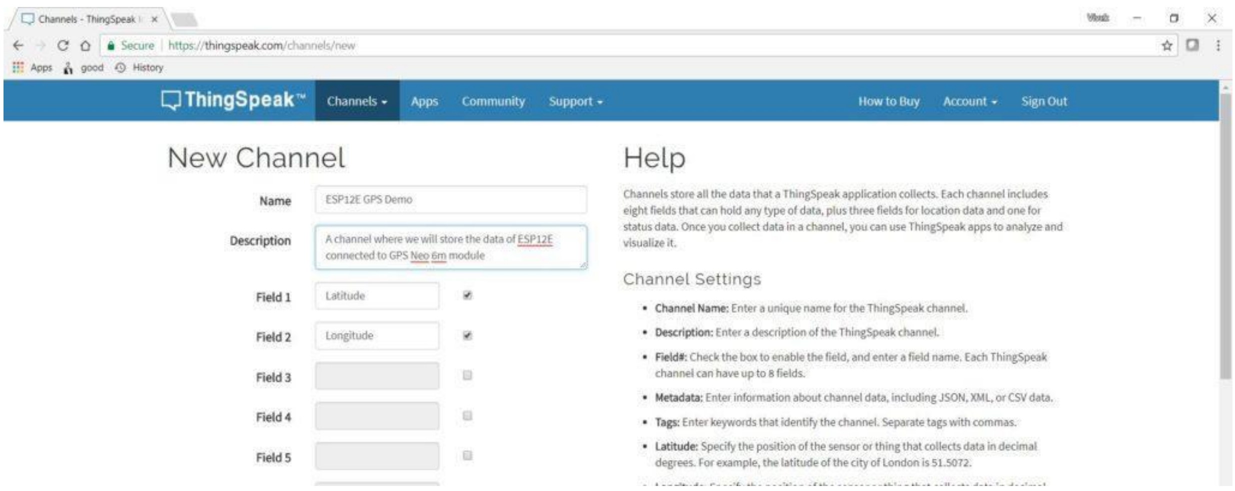
I am assuming that you already have a thingspeak account. If not then visit https://thingspeak.com/ and create your account. And follow the below process:

1. Got to "Channels" and click on "New Channel".



Collect data in a ThingSpeak channel from a device, from another channel, or from the web.

2. Then Enter following details as shown in the below image.



New Channel

Name: ESP12E GPS Demo

Description: A channel where we will store the data of ESP12E connected to GPS Neo 6m module

Field 1: Latitude ✓

Field 2: Longitude ✓

Field 3:

Field 4:

Field 5:

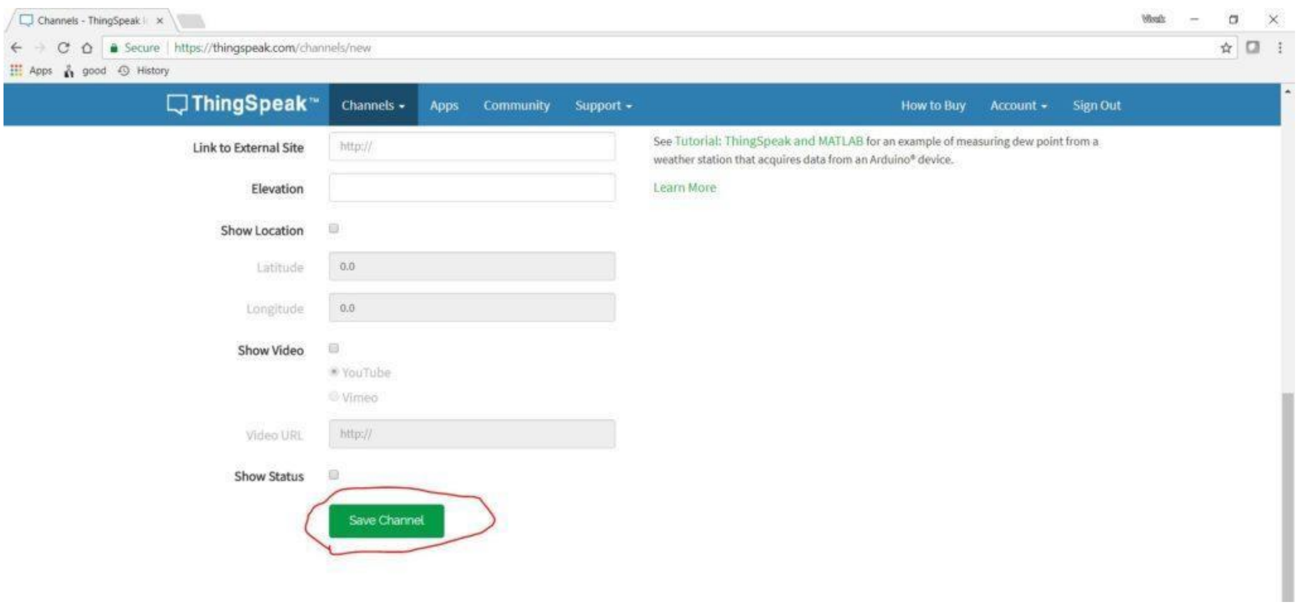
Help

Channels store all the data that a ThingSpeak application collects. Each channel includes eight fields that can hold any type of data, plus three fields for location data and one for status data. Once you collect data in a channel, you can use ThingSpeak apps to analyze and visualize it.

Channel Settings

- **Channel Name:** Enter a unique name for the ThingSpeak channel.
- **Description:** Enter a description of the ThingSpeak channel.
- **Field#:** Check the box to enable the field, and enter a field name. Each ThingSpeak channel can have up to 8 fields.
- **Metadata:** Enter information about channel data, including JSON, XML, or CSV data.
- **Tags:** Enter keywords that identify the channel. Separate tags with commas.
- **Latitude:** Specify the position of the sensor or thing that collects data in decimal degrees. For example, the latitude of the city of London is 51.5072.

3. Scroll down and hit "Save Channel" as shown in the below image.



See Tutorial: ThingSpeak and MATLAB for an example of measuring dew point from a weather station that acquires data from an Arduino® device.

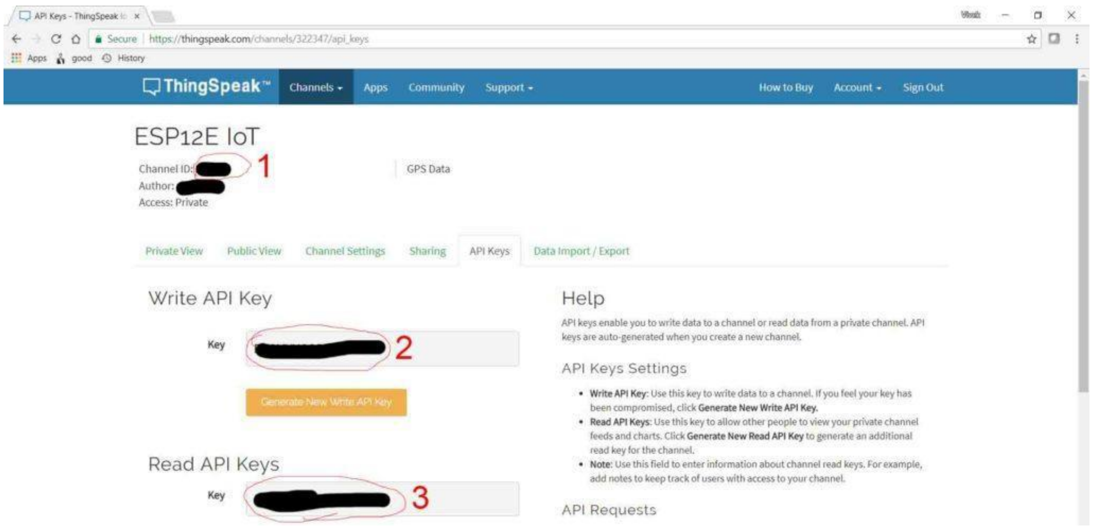
4. Now your channel has been successfully created. Click on your created channel and then click on "API Keys" tab and you will get the following details:

*Channel ID - 1

*Write API Key - 2

*Read API Key – 3

Which will be used by us for writing, reading the data on/from thingspeak server? So copy and pasted it on a notepad file and keep it ready.



Channel ID: [redacted]
Author: [redacted]
Access: Private

Write API Key: [redacted]

Read API Keys: [redacted]

Help

API keys enable you to write data to a channel or read data from a private channel. API keys are auto-generated when you create a new channel.

API Keys Settings

- **Write API Key**: Use this key to write data to a channel. If you feel your key has been compromised, click **Generate New Write API Key.**
- **Read API Keys**: Use this key to allow other people to view your private channel feeds and charts. Click **Generate New Read API Key** to generate an additional read key for the channel.
- **Note**: Use this field to enter information about channel read keys. For example, add notes to keep track of users with access to your channel.

API Requests

## Showing Thingspeak GPS data in Google Maps using Javascript and HTML

We need Google API key to link google maps and thingspeak.

**HOW DO WE CREATE GOOGLE API KEY?**

Google has recently launched the Google Maps Platform to manage the maps used on your websites.

**With this launch, the process to get Google Maps API key has changed.**

From June 11, 2018, it is mandatory to enable the billing account with a credit card/bank account details and have a valid API key for all projects.

Here is an article that shows how to enable billing account and get a new API key. Once you generate an API key, make sure to update it in plugin's Google Map settings.

**Do I need to pay anything to use a new API key?**

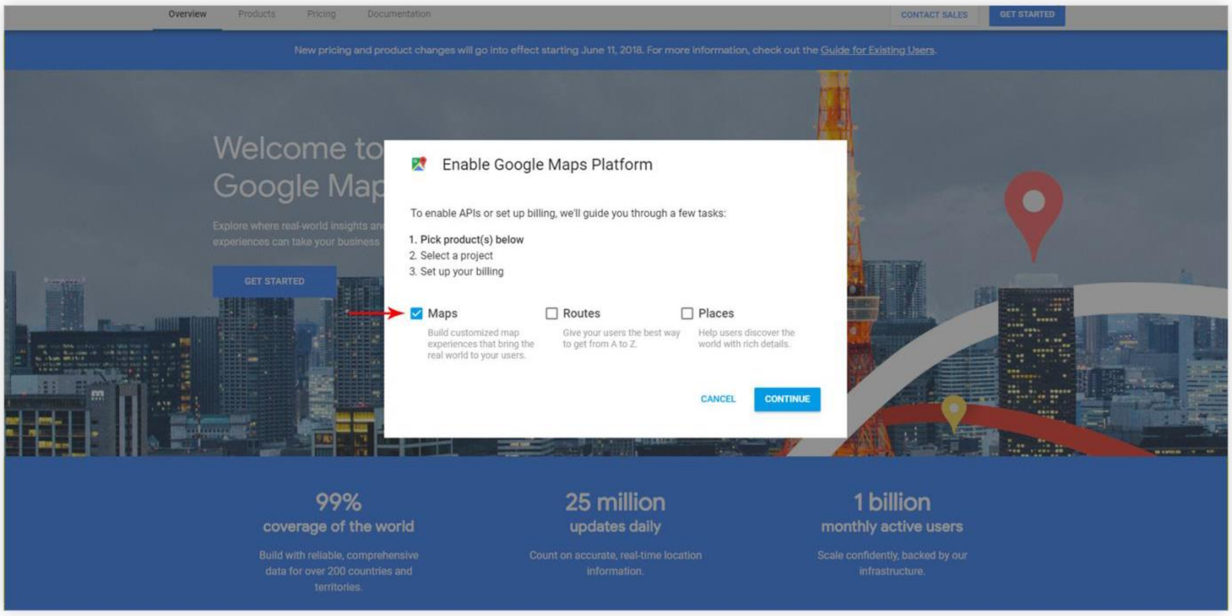
Not initially! After creating a billing account in the Google Maps Platform, you will gain access to your $200 of free monthly usage. After that, the charges will be applied as per the standard rates decided by the Google Maps Platform.

Following are the steps that will help you get an API key to use Google Maps

**Step 1** – Follow this LINK

https://developers.google.com/maps/documentation/javascript/get-api-key in order to generate your API key.

Click GET STARTED button.
Select Maps from the list and click to CONTINUE.



New pricing and product changes will go into effect starting June 11, 2018. For more information, check out the Guide for Existing Users.

Enable Google Maps Platform

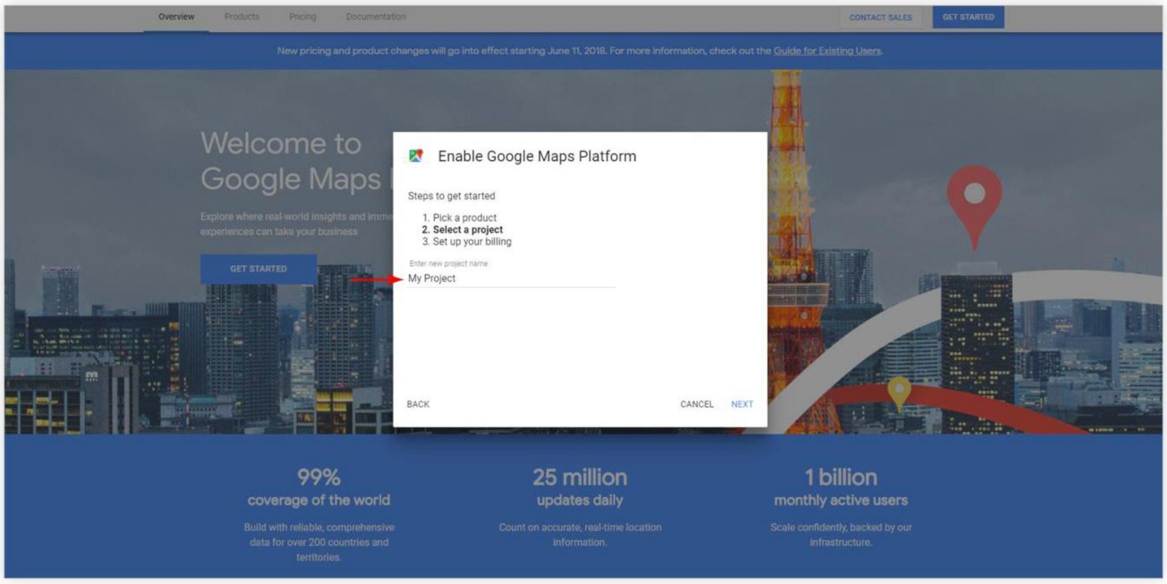
To enable APIs or set up billing, we'll guide you through a few tasks:

1. Pick product(s) below
2. Select a project
3. Set up your billing

☑ Maps — Build customized map experiences that bring the real world to your users.
☐ Routes — Give your users the best way to get from A to Z.
☐ Places — Help users discover the world with rich details.

99% coverage of the world — Build with reliable, comprehensive data for over 200 countries and territories.
25 million updates daily — Count on accurate, real-time location information.
1 billion monthly active users — Scale confidently, backed by our infrastructure.

**Step 2** – Create a new project or select one from the existing ones. Click Next.



New pricing and product changes will go into effect starting June 11, 2018. For more information, check out the Guide for Existing Users.

Enable Google Maps Platform

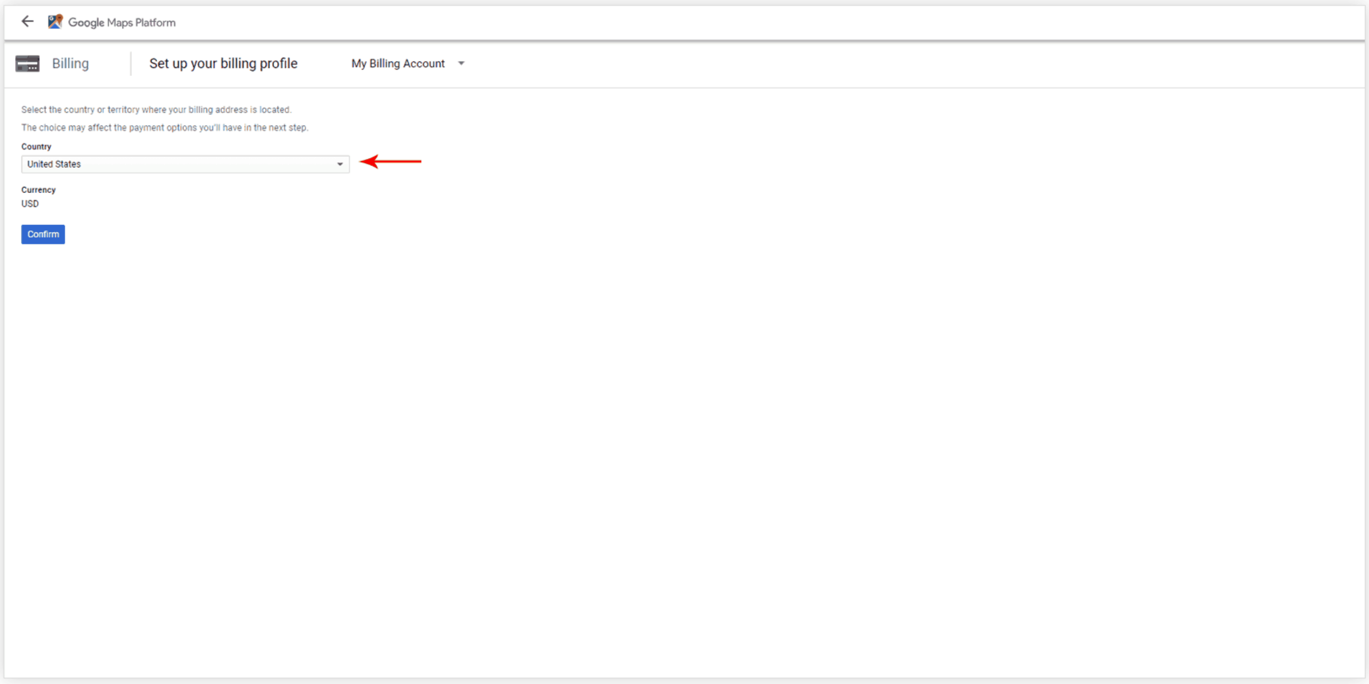
Steps to get started

1. Pick a product
2. Select a project
3. Set up your billing

Enter new project name: My Project

99% coverage of the world — Build with reliable, comprehensive data for over 200 countries and territories.
25 million updates daily — Count on accurate, real-time location information.
1 billion monthly active users — Scale confidently, backed by our infrastructure.

**Step 3** – To CREATE BILLING ACCOUNT, select the country. Click Confirm.

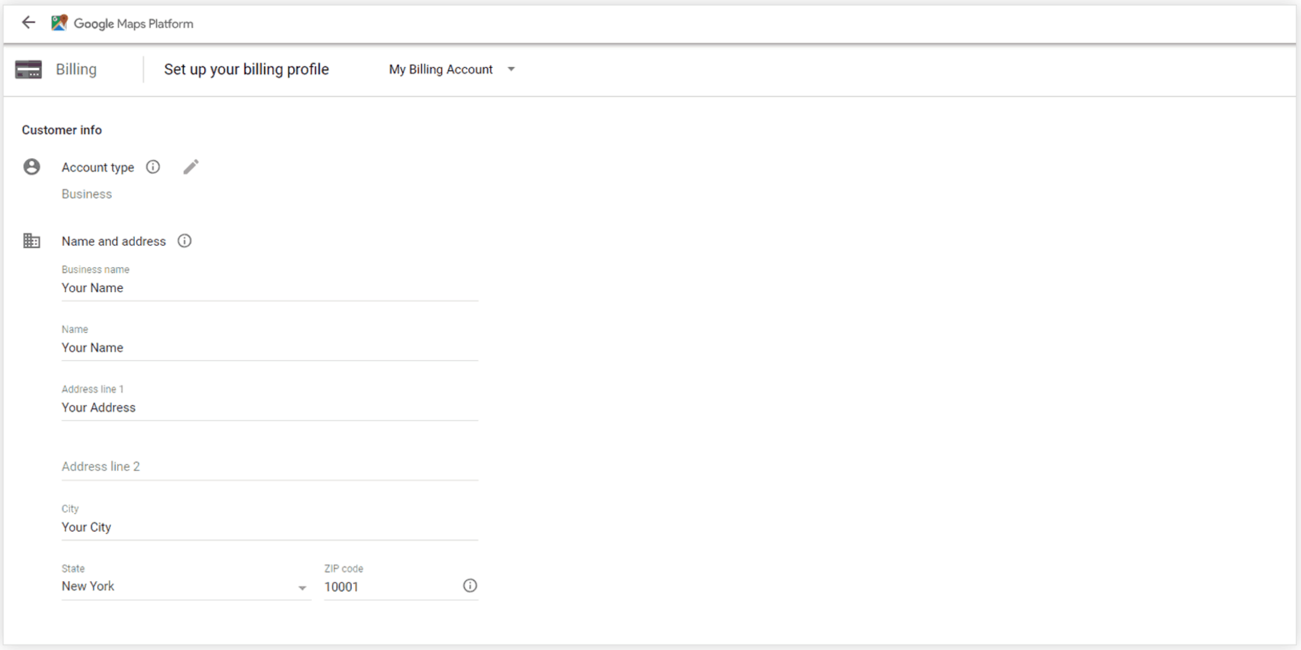


Select the country or territory where your billing address is located.
The choice may affect the payment options you'll have in the next step.

Country
United States

Currency
USD

**Step 4** – Add Customer info.



Customer info

Account type
Business

Name and address

Business name
Your Name

Name
Your Name

Address line 1
Your Address

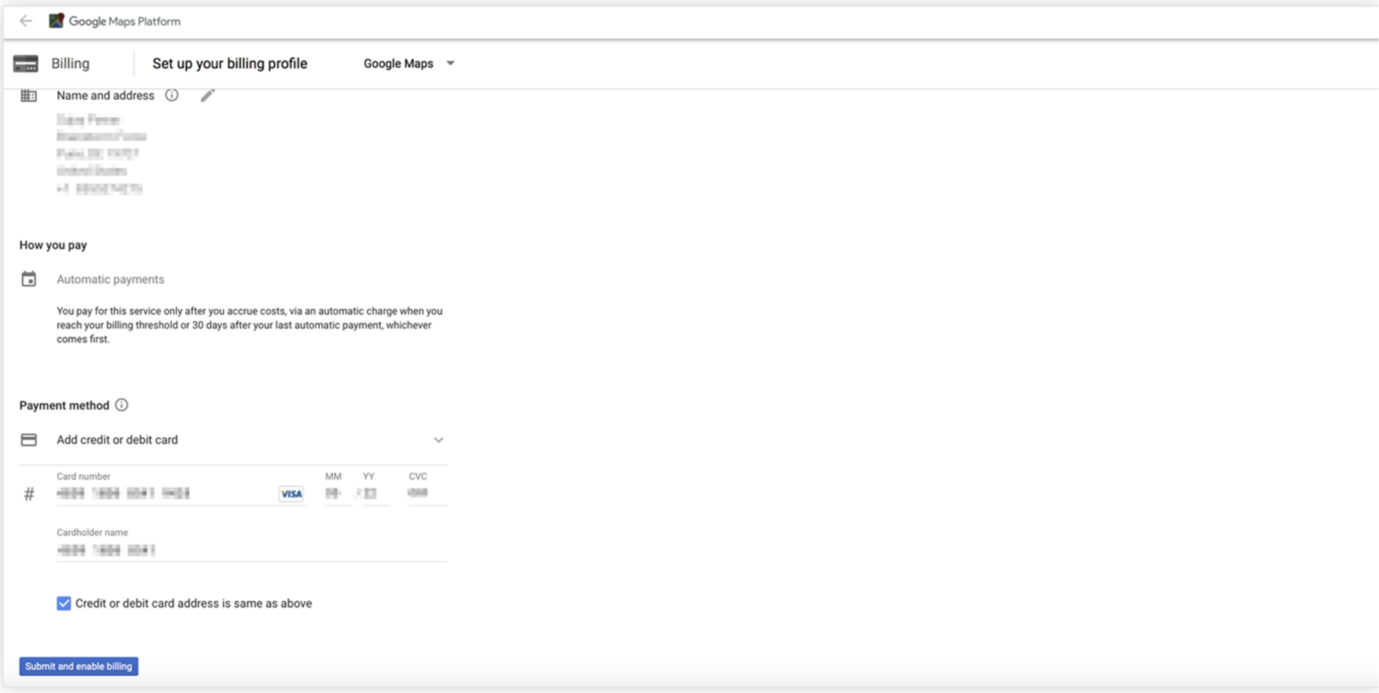
Address line 2

City
Your City

State
New York

ZIP code
10001

**Step 5** – Choose the Payment method and add payment details. Click Submit and enable billing.



Name and address

[redacted]

How you pay

Automatic payments

You pay for this service only after you accrue costs, via an automatic charge when you reach your billing threshold or 30 days after your last automatic payment, whichever comes first.

Payment method

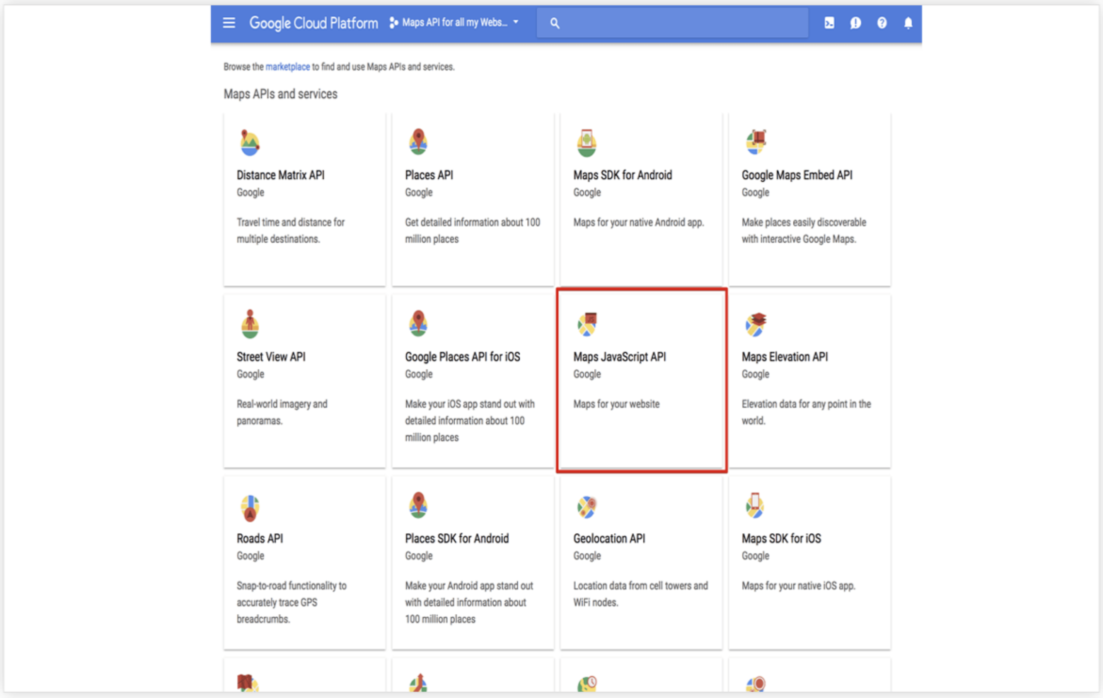
Add credit or debit card

Card number [redacted] MM [redacted] YY [redacted] CVC [redacted]

Cardholder name [redacted]

☑ Credit or debit card address is same as above

**Step 6** – Choose Maps JavaScript API



Browse the marketplace to find and use Maps APIs and services.

Maps APIs and services

Distance Matrix API
Google
Travel time and distance for multiple destinations.

Places API
Google
Get detailed information about 100 million places

Maps SDK for Android
Google
Maps for your native Android app.

Google Maps Embed API
Google
Make places easily discoverable with interactive Google Maps.

Street View API
Google
Real-world imagery and panoramas.

Google Places API for iOS
Google
Make your iOS app stand out with detailed information about 100 million places

Maps JavaScript API
Google
Maps for your website

Maps Elevation API
Google
Elevation data for any point in the world.

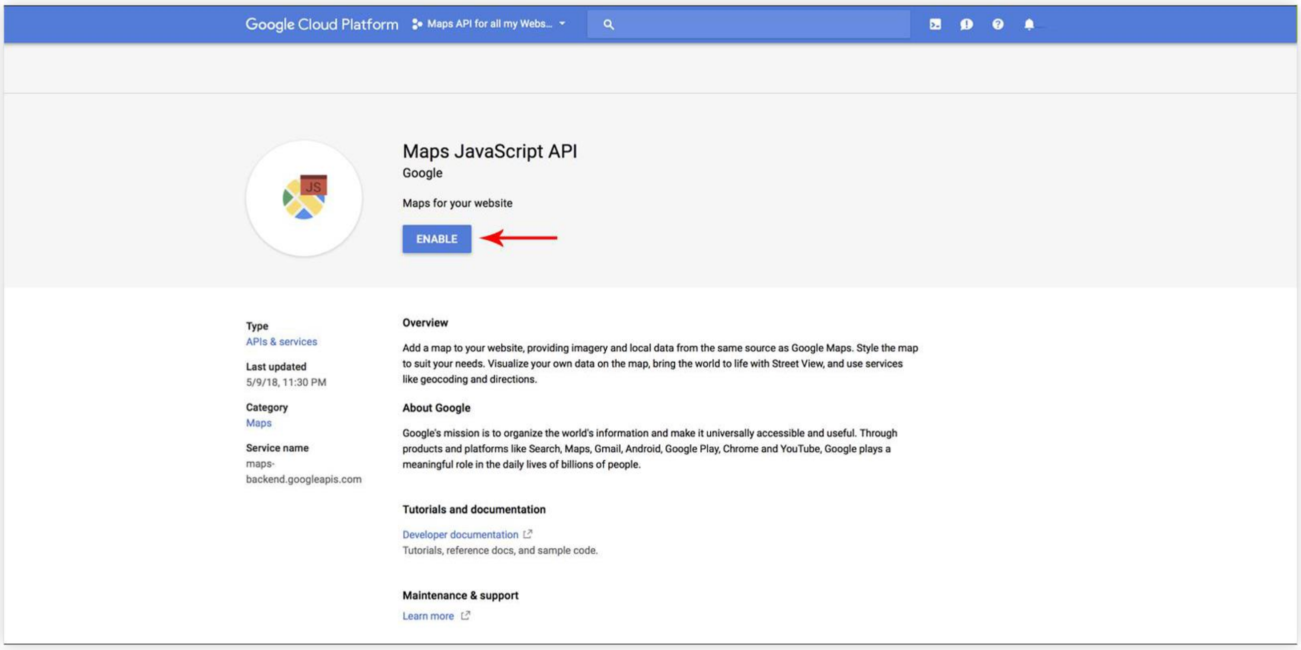
Roads API
Google
Snap-to-road functionality to accurately trace GPS breadcrumbs.

Places SDK for Android
Google
Make your Android app stand out with detailed information about 100 million places

Geolocation API
Google
Location data from cell towers and WiFi nodes.

Maps SDK for iOS
Google
Maps for your native iOS app.

**Step 7** – Enable maps for your website.



Maps JavaScript API

Google

Maps for your website

Type
APIs & services

Last updated
5/9/18, 11:30 PM

Category
Maps

Service name
maps-backend.googleapis.com

Overview

Add a map to your website, providing imagery and local data from the same source as Google Maps. Style the map to suit your needs. Visualize your own data on the map, bring the world to life with Street View, and use services like geocoding and directions.

About Google

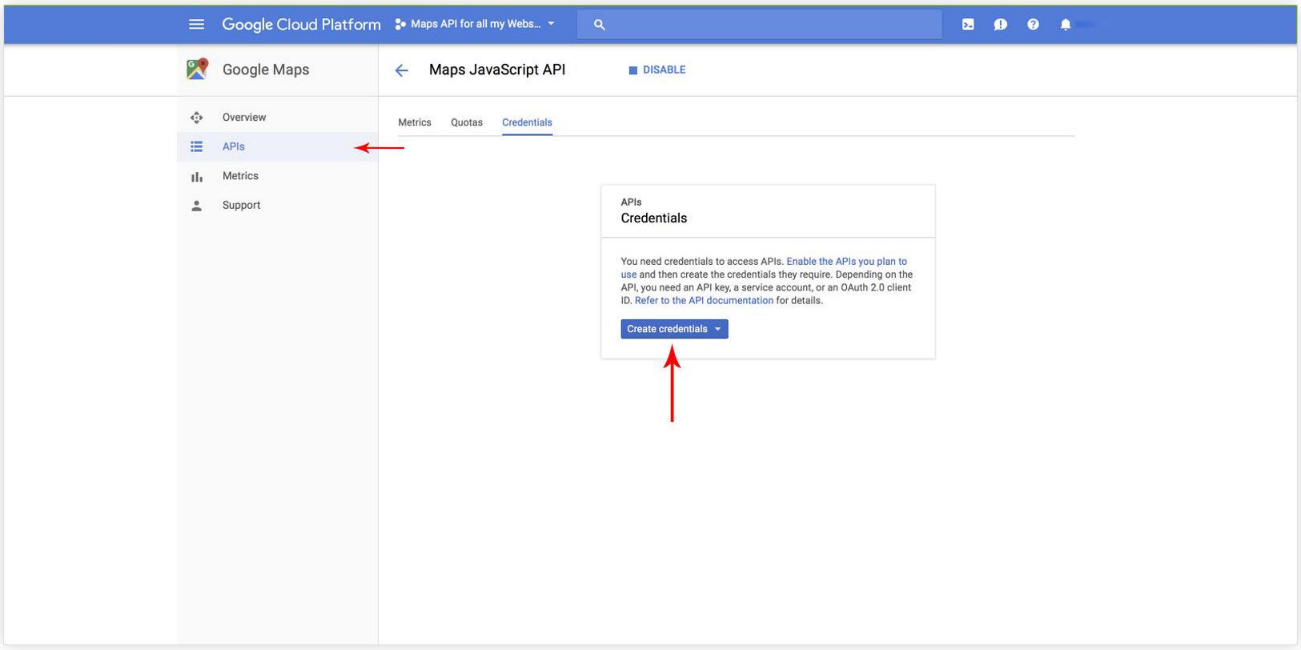
Google's mission is to organize the world's information and make it universally accessible and useful. Through products and platforms like Search, Maps, Gmail, Android, Google Play, Chrome and YouTube, Google plays a meaningful role in the daily lives of billions of people.

Tutorials and documentation

Developer documentation
Tutorials, reference docs, and sample code.

Maintenance & support
Learn more

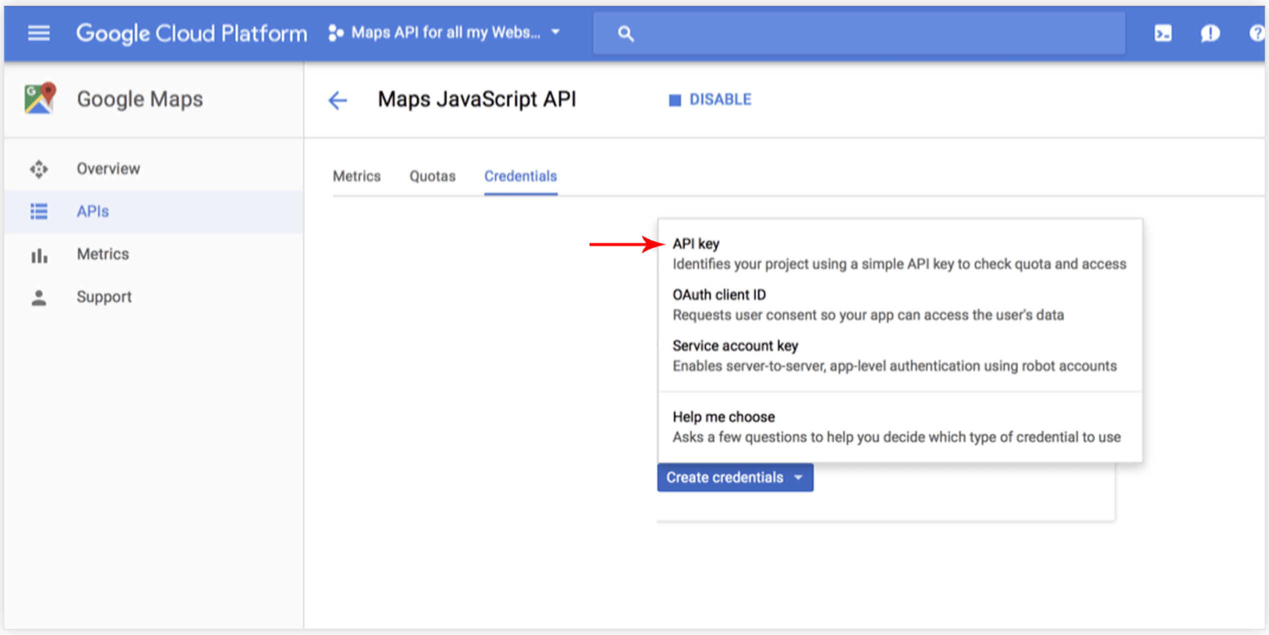
**Step 8** – Choose APIs credentials.



APIs
Credentials

You need credentials to access APIs. Enable the APIs you plan to use and then create the credentials they require. Depending on the API, you need an API key, a service account, or an OAuth 2.0 client ID. Refer to the API documentation for details.

**Step 9** – Select API Key option.



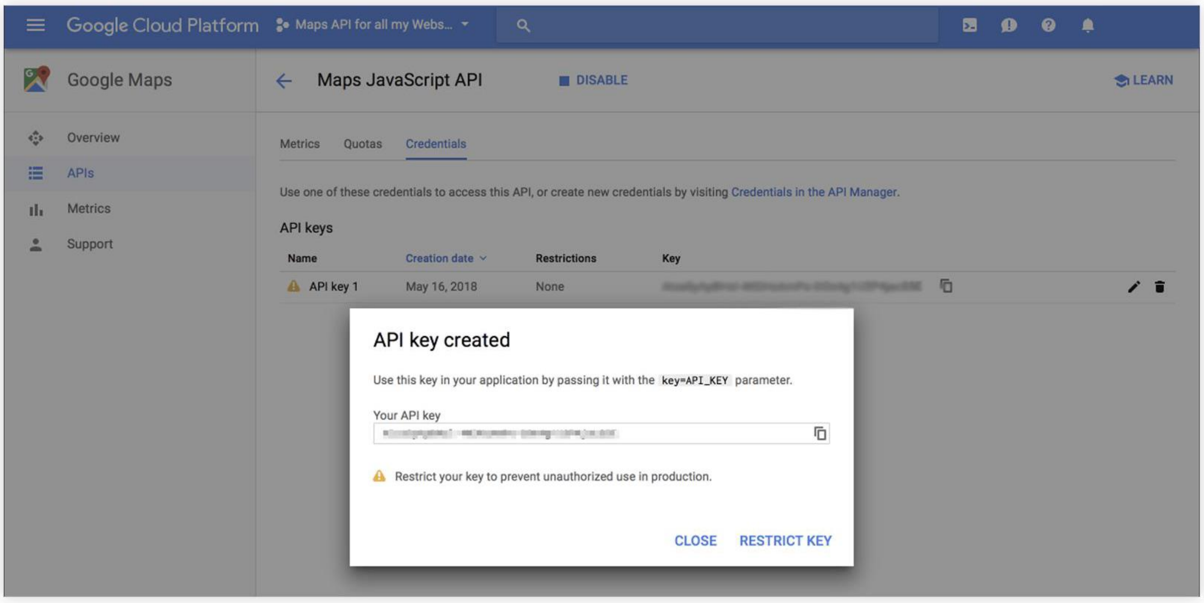
API key
Identifies your project using a simple API key to check quota and access

OAuth client ID
Requests user consent so your app can access the user's data

Service account key
Enables server-to-server, app-level authentication using robot accounts

Help me choose
Asks a few questions to help you decide which type of credential to use

**Step 10** – Copy the generated API key.



Use one of these credentials to access this API, or create new credentials by visiting Credentials in the API Manager.

API keys

| Name | Creation date | Restrictions | Key |
|---|---|---|---|
| API key 1 | May 16, 2018 | None | [illegible] |

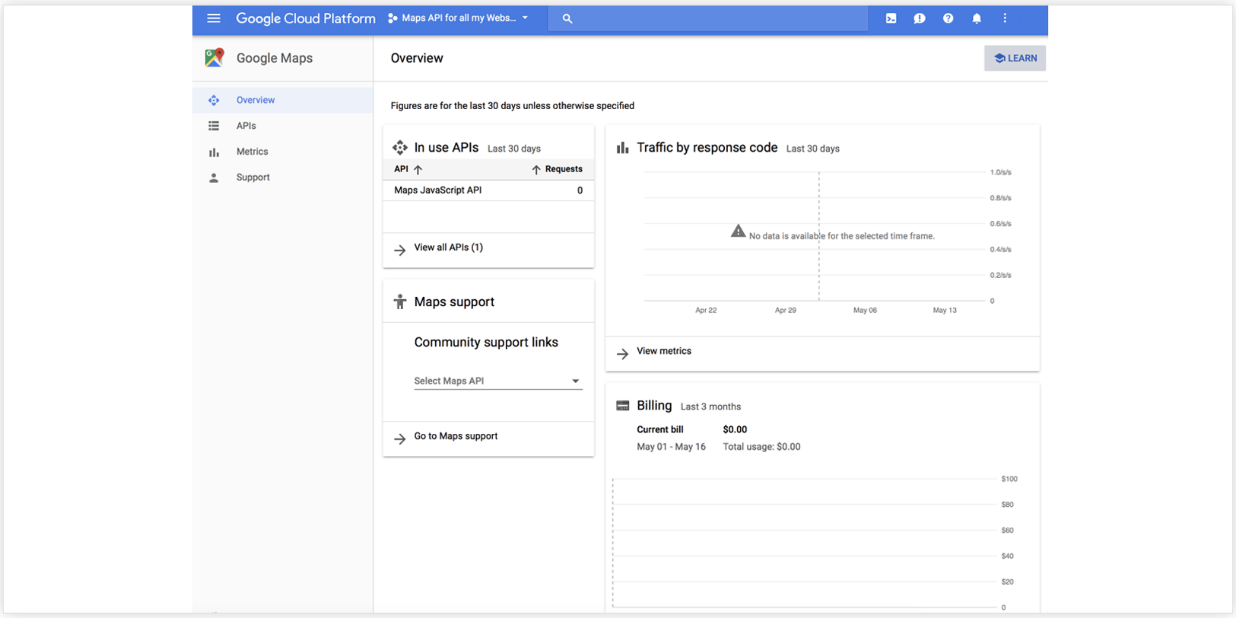
API key created

Use this key in your application by passing it with the `key=API_KEY` parameter.

Your API key

Restrict your key to prevent unauthorized use in production.

**Step 11**– You can see map usage on your account overview page.



After getting GOOGLE API KEY, we need to write  html code to read values from thingspeak and display on Google maps.

**CODE TO READ DATA FROM THINGSPEAK AND DISPLAYON GOOGLE MAPS:-**

```
<!DOCTYPE html>

<html>

 <head>

   <title>VECHILE TRACKING USING IOT</title>

   <meta name="viewport" content="initial-scale=1.0, user-scalable=no">

   <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: 25px;

      padding: 0;

    }

  </style>

 <script async defer
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyBekTHC3mQ9Am0xgL_2k9PJJH7t
35_Z5cw&callback=initMap">

     </script>

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>

  <script>

 /*TO READ DATA FROM THINGSPEAK*/

    var map;

      var x;

      function loadmaps(){

          $.getJSON("https://api.thingspeak.com/channels/THINGSPEAK CHANNEL
ID/fields/1/last.json?api_key=THINGSPEAK READ API KEY&results=2", function(result){


/*JSON DATA IS STORED IN "result"*/

          var m = result;
```

```
        x=Number(m.field1); /*EXTRACTING FIELD DATA*/

                var xv=x;

$(document).ready(function() {

   $('#Latitude').val(xv);

});

 });

        $.getJSON("https://api.thingspeak.com/channels/THINGSPEAK CHANNEL
ID/fields/2/last.json?api_key=THINGSPEAK READ API KEY&results=2", function(result1){

         /*JSON DATA IS STORED IN “result1”*/

        var m = result1;

        y=Number(m.field2);  /*EXTRACTING FIELD DATA*/

         var yv=y;

          $(document).ready(function() {

   $('#Longitude').val(yv);

});

      }).done(function() {

             initialize();

   });

      }

/*TO RELOAD THE PAGE AFTER EVERY 9 SECONDS*/

      window.setInterval(function(){

      loadmaps();
      }, 9000);
   function initialize() {
      //alert(y);
```

**/*TO LOCATE LATITUDE AND LONGITUDE ON GOOGLE MAPS*/**

```
     var mapOptions = {
       zoom: 17,
       center: {lat: x, lng: y}
     };
     map = new google.maps.Map(document.getElementById('map'),
         mapOptions);
```

**/* TO SHOW MARKER ON GOOGLE MAPS*/**

```
     var marker = new google.maps.Marker({
       position: {lat: x, lng: y},
       map: map
     });
     var infowindow = new google.maps.InfoWindow({
       content: '<p>Marker Location:' + marker.getPosition() + '</p>'
     });
     google.maps.event.addListener(marker, 'click', function() {
       infowindow.open(map, marker);
     });
   }
```

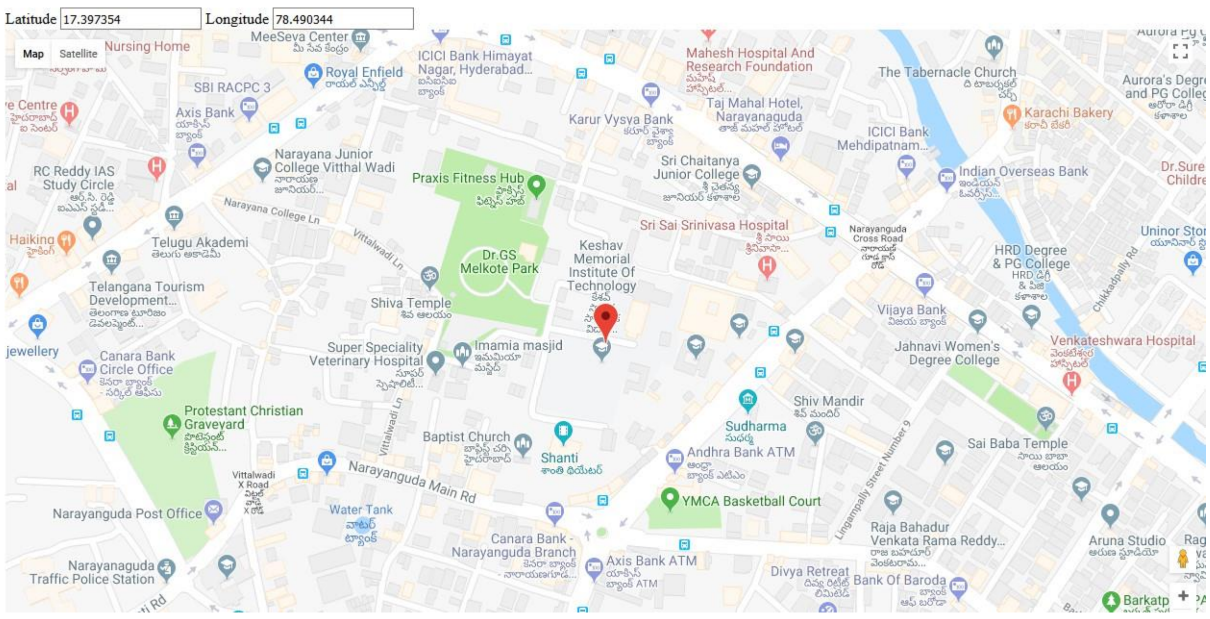
```
     google.maps.event.addDomListener(window,  'load',  initialize);

   </script>

  </head>

  <body >

   <p1>Latitude</p1>

   <input id="Latitude" type="text">

   <p1>Longitude</p1>

   <input id="Longitude" type="text">

    <div id="map"></div>

  </body>

</html>
```

**Replace the following credentials:**

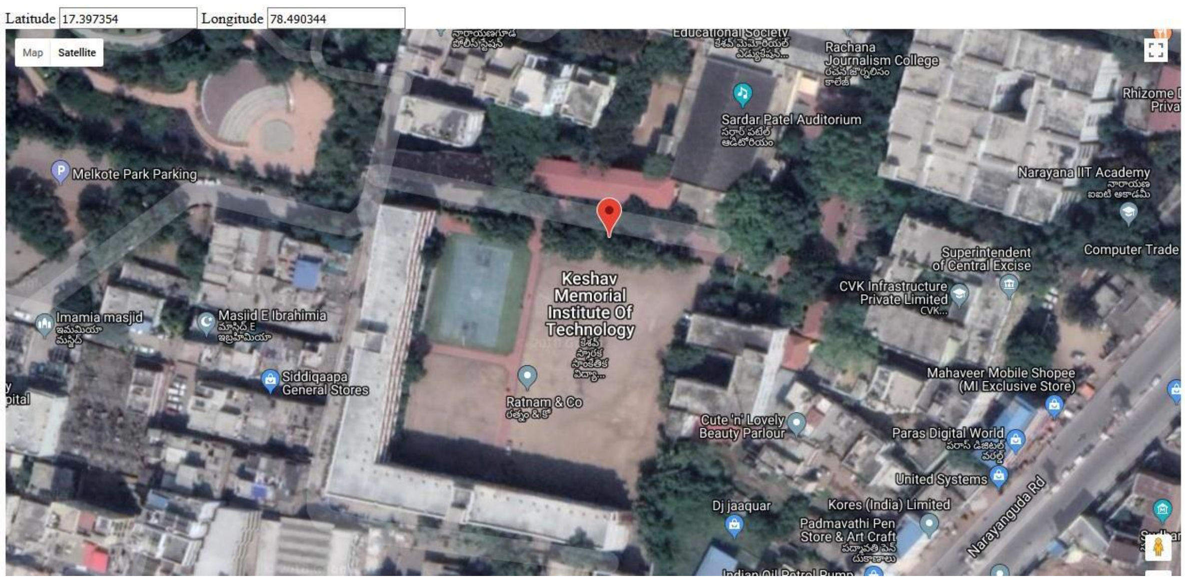
1. Your copied, google api key
2. Your Thingspeak Read API key
3. Your Thingspeak Read API Key
4. Your Thingspeak Channel ID
5. Your Thingspeak Channel ID

Save this code with **.html** extension.

Now double click on your webpage for example “Vehicle tracker.html” and wait for some time. It will take some time to load the map and marker. If you have done everything right, then you will get the following output.


Latitude 17.397354 Longitude 78.490344

If you select satellite view, then you will get the following output.


Latitude 17.397354 Longitude 78.490344

# Chapter 07

# ADVANTAGES & FUTURE SCOPE

## 7.1 ADVANTAGES:

Commercial fleet operators are by far the largest users of vehicle tracking systems. These systems are used for operational functions such as routing, security, dispatch and collecting on-board information.

These are also used for fire detector in large vehicles like train, bus etc. because the vehicle like train contains large number of people and the sending alert of fire accident can save many lives.

The applications for this project are in military, navigation, automobiles, aircrafts, fleet management, remote monitoring, remote control, security systems, tele services, etc.

- Fleet monitoring
- Vehicle scheduling
- Route monitoring
- Driver monitoring
- Accident analysis
- Geo-fencing geo-coding

These are just a few advantages of the project that has been introduced in this report. We can interface more number of sensors in order to serve multiple purposes. The microcontroller that has been used in this project have inbuilt ADCs and hence the controller is capable of accepting analog inputs, which is the biggest advantage. Since all real world signals are analog in nature, by incorporating different sensors required purpose can be served.

## 7.2 FUTURE SCOPE:

- We can use the EEPROM to store the previous Navigating positions up to 256 locations and we can navigate up to N number of locations by increasing its memory.
- We can reduce the size of the kit by using GPS+GSM on the same module.

- We can increase the accuracy up to 3m by increasing the cost of the GPS receivers.
- We can use our kit for detection of bomb by connecting to the bomb detector.
- With the help of high sensitivity vibration sensors we can detect the accident.

Whenever vehicle unexpectedly had an accident on the road with help of vibration sensor we can detect the accident and we can send the location tothe owner, hospital and police.

- We can use our kit to assist the traffic. By keeping the kits in the entire vehicles and by knowing the locations of all the vehicles.
- If anybody steals our car we can easily find our car around the globe. By keeping vehicle positioning vehicle on the vehicle.

# Chapter 08

# Conclusion

Vehicle tracking system makes better fleet management and which in turn brings large profits. Better scheduling or route planning can enable you handle larger jobs loads within a particular time. Vehicle tracking both in case of personal as well as business purpose improves safety and security, communication medium, performance monitoring and increases productivity. So in the coming year, it is going to play a major role in our day-to-day living.

Main motto of the project is to incorporate different types of sensors so that they help in decrease the chances of losing life in such accident which we can’t stop from occurring. Whenever accident is alerted the paramedics are reached to the particular location to increase the chances of life. This device invention is much more useful for the accidents occurred in deserted places and midnights. This vehicle tracking and accident alert feature plays much more important role in day to day life in future.