INDEX

INTRODUCTION

Vehicle Tracking System is a system which will track the live position of vehicle with the help of satellites. Here we tried to reproduce the same, with all the software and the hardware support. In general, tracking of vehicles is a process in which we request the location in form of **latitude and longitude (Geographic Coordinates)** and send them to the server. Geographic Coordinates can pin point any location on Earth. This system is very efficient for outdoor application purposes, having accuracy of about 5m. These devices are widely used in tracking passenger vehicles, stolen vehicles, school/college buses, etc. They are very commonly used in fleet management and asset tracking applications. Today these systems can not only track the location of the vehicle but can also report the speed and locate the vehicle on the map. They can be used for multiple vehicles simultaneously and can be used from anywhere in world with the help of interne

HARDWARE USED

- GSM module

  A GSM modem or GSM module is a hardware device that uses GSM mobile telephone technology to provide a data link to a remote network. From the view of the mobile phone network, they are essentially identical to an ordinary mobile phone, including the need for a SIM to identify themselves to the network. GSM modems typically provide TTL-level serial interface to their host. They are usually used as part of an embedded system. They communicate using AT commands.

  Use-

  Used to transmit data from vehicle to server.

- GPS module

  The Global Positioning System (GPS) is the most widely used satellite navigation system around the world. It is one of the Global Navigation Satellite Systems (GNSS) that provides geolocation, time, and velocity information. The GPS receiver gets a signal from each GPS satellite. The satellites transmit the exact time the signals are sent. By subtracting the time, the signal was transmitted from the time it was received, the GPS can tell how far it is from each satellite. The GPS receiver also knows the exact position in the sky of the satellites, at the moment they sent their signals. So given the

travel time of the GPS signals from three satellites and their exact position in the sky, the GPS receiver can determine your position in three dimensions – latitude, longitude and altitude.

Use-

Used to know the geographic coordinates of the vehicle

- Microcontroller (DIP package) –ATmega328p

The ATmega328 is a single-chip microcontroller created by Atmel in the mega vr family. It has a modified Harvard architecture processor core. The Atmel 8-BIT AVR RISC -based microcontroller combines 32 KB ISP Flash memory with read-while-write capabilities, 1 KB EEPROM, 2 KB SRAM, 23 general purpose I/O lines, 32 general purpose working register three flexible timer/counters with compare modes, internal and external interrupts, serial programmable USRAT, a byte-oriented 2-wire serial interface, SPI serial port, 6-channel 10-bit A/D Converter (8-channels in TQFP and MFN packages), programmable watchdog timer with internal oscillator, and five software selectable power saving modes. The device operates between 1.8-5.5 volts. The device achieves throughput approaching 1 MIPS per MHz

Use-

To take data from GPS module, parse it and then transfer that data to GSM module using proper AT commands

- CRYSTAL 16MHZ

  It provides the clock signal for the microcontroller in the Arduino to work. Clock signal is the heartbeat that makes any micro controller or microprocessor do its work.

- LED
- RESISTOR
- CAPACITOR

SOFTWARE USED

Flask

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries.[2] It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. It is used for server.

Proteus

The Proteus Design Suite is a proprietary software tool suite used primarily for electronic design automation. It is a Windows application for schematic capture, simulation, and PCB (Printed Circuit Board) layout design. It can be purchased (it's a paid software) in many configurations, depending on the size of designs being produced and the requirements for microcontroller simulation.

- Schematic Capture

  Schematic capture in the Proteus Design Suite is used for both the simulation of designs and as the design phase of a PCB layout project. It is therefore a core component and is included with all product configurations.

- Microcontroller Simulation

  The micro-controller simulation in Proteus works by applying either a hex file or a debug file to the microcontroller part on the schematic. It is then co-simulated along with any analog and digital electronics connected to it.

This enables its use in a broad spectrum of project prototyping in areas such as motor control, temperature control and user interface design. It also finds use in the general hobbyist community and, since no hardware is required, is convenient to use as a training or teaching tool.

- PCB Design

  The PCB Layout module is automatically given connectivity information in the form of a netlist from the schematic capture module. It applies this information, together with the user specified design rules and various design automation tools, to assist with error free board design. PCB's of up to 16 copper layers can be produced with design size limited by product configuration.

- 3D Verification

  The 3D Viewer module allows the board under development to be viewed in 3D together with a semi-transparent height plane that represents the boards enclosure. STEP output can then be used to transfer to mechanical CAD software such as Solidworks or Autodesk for accurate mounting and positioning of the board.

Arduino (IDE)

The Arduino Integrated Development Environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in functions from C and C++. It is used to write and upload programs to Arduino compatible boards, but also, with the help of third-party cores, other vendor development boards and microcontrollers.

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. By default, avrdude is used as the uploading tool to flash the user code onto official Arduino boards.

With the rising popularity of Arduino as a software platform, other vendors started to implement custom open-source compilers and tools (cores) that can build and upload sketches to other microcontrollers that are not supported by Arduino's official line of microcontrollers.

Here, we are using this software for code compilation and HEX file generation.

PYCHARM (IDE)

PyCharm is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCS), and supports web development with Django as well as data science with Anaconda.

PyCharm is cross-platform, with Windows, macOS and Linux versions. The Community Edition is released under the Apache License, and there is also Professional Edition with extra features – released under a proprietary license. Some features-

- Coding assistance and analysis, with code completion, syntax and error highlighting, linter integration, and quick fixes
- Project and code navigation: specialized project views, file structure views and quick jumping between files, classes, methods and usages
- Python refactoring: includes rename, extract method, introduce variable, introduce constant, pull up, push down and others
- Support for web frameworks: Django, web2py and Flask
- Integrated Python debugger
- Integrated unit testing, with line-by-line code coverage
- Version control integration: unified user interface for Mercurial, Git, Subversion, Perforce and CVS with change lists and merge.

WORKING PRINCIPLE

This project is divided mainly in 2 parts and one user interface as website to access to information:

1. Hardware
2. Server

HARDWARE

This hardware will be installed on the vehicle, it consists of 3 major modules/components – MCU(ATmega328p), GSM module and GPS module.

Here the main module which gives the location id GPS module, which gives the location of itself in the form of NMEA string. GPS module uses many satellites to determine its geographical coordinates. It achieves this by receiving signal from at least 4 satellites, taking in account the time shift, theory of relativity etc and voila, we get our accurate location on Earth. GPS module doesn't need initialization to be done, it just starts sending data as soon as it gets power.

Then the location in the form of NMEA string is transmitted to MCU using UART communication protocol along with several other information, so there our MCU, parse that string and then extracts the useful information from it , that is longitude and longitude and send them to GSM module using AT commands. The working commands for our MCU is written in Arduino.

GSM module uses SIM for its unique identification in world and that SIM allows us to use GPRS services for our project. GSM module after being properly initialized, we send it commands to access internet and open certain links or in other words send certain commands to our server for it to respond. GSM module also uses UART communication protocol to communicate with MCU and GSM module also replies to the AT commands it's been sent, to tell their proper status.

Here we first send the command to initialize the GSM module and provide the APN (Access Point Name) then send command to connect to certain IP address with certain Port number and certain protocol and then we start sending data to that particular IP address, along with the length oof data being sent.

For power of this hardware, we can either connect it to vehicle battery or connect some external battery.

SOFTWARE

For server we used, Flask, which is web framework based on Python. Our server has two address '/' and '/add' where first one denotes the homepage where we can see all the vehicles connected to that server and their past running information, while the other is not for general use, it is meant for the hardware it open , it accepts 4 arguments namely-

- lat – latitude of vehicle
- long – longitude of vehicle
- id – id of the vehicle
- key – private key of the vehicle so that none else can update its location

On successful update it returns with the string containing the longitude and latitude, just posted to confirm the value, in case of any error or any parameter value is missing or is not of correct format or key is wrong then it shows the respective error in string format.

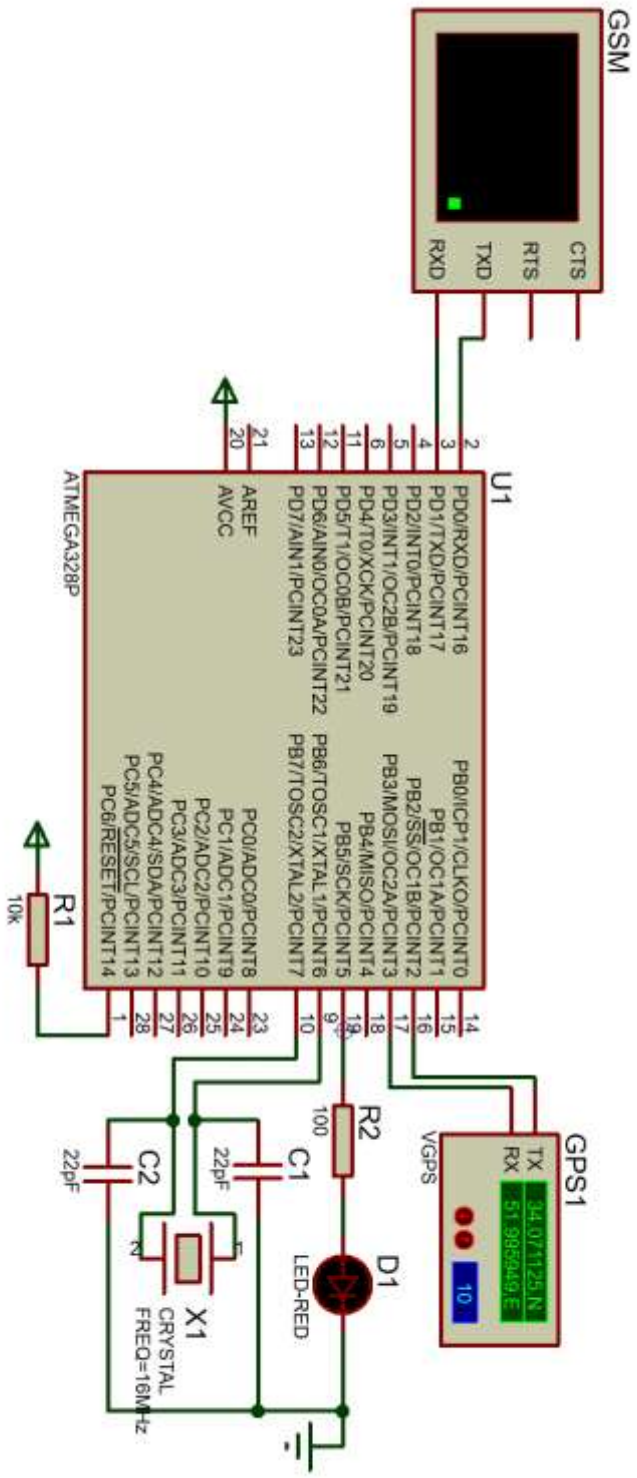
It stores the data received from vehicles in a csv file, consisting of 4 columns namely time, latitude, longitude, user_id.

Whenever a user requests to view homepage arrives, it takes that csv data and makes a new '.js' file (JavaScript) to use it into the webpage. The webpage then uses the data in JavaScript file dictionary with Google Maps API to load map, mark the pins for different user, mark the line for path taken by different user with different colour and set the opening coordinates of maps in that zoom level. Here Google Maps API takes the coordinates in decimal degree format whereas the vehicle GPS module send it in degree decimal minute format, so it's the job of server to convert the format from one to another.

FINAL USER INTERFACE

As you know in this world what happens behind the door doesn't matter, as a user we expect simple things, so for user the work is very simple just connect the device to the vehicle and its battery and whenever user want to track they can open the webpage, where they can see the path of their vehicle in certain colour, set by them with the last location will have pin/marker on it to denote that this is the last updated location of this user. To update the map with current location we just need to refresh the page and it will show the updated map.

SCHEMATIC DIAGRAM



SOURCE CODE

For Microcontroller

```
#include <SoftwareSerial.h>
SoftwareSerial gps(10,11);

char str[70];
String gpsString="";
char *test="$GPGGA";
String latitude="  ", longitude="    ", key="aawaaraa";
int i, device_id=1;
boolean gps_status=0;

void setup(){
  Serial.begin(9600);
  gps.begin(9600);
  gsm_init();
  get_gps();
  delay(500);
}

void loop(){
    get_gps();
    tracking();
}

void gpsEvent(){
  gpsString="";
  while(!gps_status)
  {
     while (gps.available()>0)
     {
       char inChar = (char)gps.read();
       gpsString+= inChar;
       i++;
       if (i < 7 && gpsString[i-1] != test[i-1]){
          i=0;
          gpsString="";
       }
       if(inChar=='\r')
       if(i>65){
         gps_status=1;
         break;
```

```
        }
        else            i=0;
      }
    }
}

void get_gps(){
    gps_status=0;
    int x=0;
    while(gps_status==0)
    {
     gpsEvent();
     int str_lenth=i;
     latitude="";
     longitude="";
     int comma=0;
     while(x<str_lenth)
     {
       if(gpsString[x]==',')        comma++;
       if(comma==2)                 latitude+=gpsString[x+1];
       else if(comma==3)            latitude+=gpsString[x+1];
       else if(comma==4)            longitude+=gpsString[x+1];
       else if(comma==5)            longitude+=gpsString[x+1];
       x++;
     }
     latitude = latitude.substring(0, latitude.length()-3) +
  latitude[latitude.length()-2];
     longitude = longitude.substring(0, longitude.length()-3)
  + longitude[longitude.length()-2];
     i=0;x=0;
     delay(1000);
    }
}

void gsm_init(){
  connectGSM("AT","OK");
  connectGSM("ATE0","OK");
  connectGSM("AT+CPIN?","READY");
  connectGSM("AT+CIPSHUT","OK");
  connectGSM("AT+CGATT=1","OK");
  connectGSM("AT+CSTT=\"airtelgprs.com\",\"\",\"\"","OK");
  connectGSM("AT+CIICR","OK");
}

void connectGSM (String cmd, char *res){
  while(1){
    Serial.println(cmd);
    delay(500);
    while(Serial.available()>0)
```

```
88.        if(Serial.find(res)){
89.          delay(1000);
90.          return;
91.        }
92.      delay(500);
93.    }
94.  }
95.
96.  void tracking(){
97.      connectGSM("AT+CIPSTART=\"TCP\",\"127.0.0.1\",5000",
   "CONNECT");
98.      String url = "GET /add?lat=" + latitude + "&long=" +
   longitude + "&key=" + key + "&id=" + device_id + "
   HTTP/1.0\r\n\r\n";
99.      String start = "AT+CIPSEND=";
100.     start += url.length();
101.     Serial.println(start);
102.     Serial.println(url);
103.     Serial.write(0x1A);
104.     delay(1000);
105. }
106.
```

For Server-

```
1. from flask import Flask, render_template, request
2. import time
3. import csv
4.
5. app = Flask(__name__, template_folder='templates')
6. app.config['SEND_FILE_MAX_AGE_DEFAULT'] = 0
7. fieldnames = ['time', 'lat', 'long', 'id']
8. lat = ''
9. long = ''
10.
11.
12. def save_csv(uid):
13.     global lat
14.     global long
15.     if len(lat.split('.')[0]) != 4 or len(long.split('.')[0])
   != 5:
16.         return False
17.
18.     if lat[-1] == 'S':
19.         lat = '-' + lat
20.     elif lat[-1] == 'N':
21.         lat = '+' + lat
22.
23.     if long[-1] == 'W':
24.         long = '-' + long
25.     elif long[-1] == 'E':
26.         long = '+' + long
27.
28.     lat = lat[:-1]
29.     long = long[:-1]
30.     with open('data/data.csv', 'a', newline='') as f:
31.         csv_writer = csv.DictWriter(f, fieldnames=fieldnames)
32.         f.seek(0, 2)
33.         if f.tell() == 0:
34.             csv_writer.writeheader()
35.         csv_writer.writerow({"time": time.strftime('%d-%m
   %H:%M'), "lat": lat, 'long': long, 'id': uid})
36.     return True
37.
38.
39. def read_data(file):
40.     with open(f'data/{file}.csv', 'r') as f:
41.         if file == 'data':
42.             data = list(csv.reader(f))[1:]
43.         elif file == 'user':
44.             user_f = csv.DictReader(f)
```

```
45.             data = {}
46.             for i in user_f:
47.                 data[i['id']] = {'name': i['name'], 'color':
   i['color']}
48.     return data
49.
50.
51. def update_data():
52.     users_full = read_data('user')
53.     data = read_data('data')
54.     fdata = {}
55.     for i in data:
56.         try:
57.             fdata.__getitem__(users_full[i[3]]['name'])
58.         except KeyError:
59.             fdata[users_full[i[3]]['name']] = {'data': [],
   'color': users_full[i[3]]['color']}
60.         finally:
61.
   fdata[users_full[i[3]]['name']]['data'].append([i[0],
62.
   float(i[1][:3] + '.' + str(float(i[1][3:]) / 60)[2:8]),
63.
   float(i[2][:4] + '.' + str(float(i[2][4:]) / 60)[2:8])])
64.     last = [float(data[-1][1][:3] + '.' + str(float(data[-
   1][1][3:]) / 60)[2:8]),
65.             float(data[-1][2][:4] + '.' + str(float(data[-
   1][2][4:]) / 60)[2:8])]
66.     with open('static/where.js', 'w') as f:
67.         f.write(f'data = {str(fdata)};\nlast = {str(last)}')
68.     return
69.
70.
71. @app.route('/')
72. def hello_world():
73.     update_data()
74.     data = read_data('data')
75.     lat_temp = float(data[-1][1][:3] + '.' + str(float(data[-
   1][1][3:]) / 60)[2:])
76.     long_temp = float(data[-1][2][:4] + '.' +
   str(float(data[-1][2][4:]) / 60)[2:])
77.     return render_template('index.html',
   lat=str(lat_temp)[:10] + '°', long=str(long_temp)[:10] + '°')
78.
79.
80. @app.route('/add')
81. def add_data():
82.     global lat
83.     global long
```

```
84.     lat = request.args.get('lat', default='', type=str)
85.     long = request.args.get('long', default='', type=str)
86.     password = request.args.get('key', default='', type=str)
87.     uid = request.args.get('id', default=-1, type=int)
88.     if password == '' or lat == '' or long == '' or uid == -
   1:
89.         return "Missing argument(s)"
90.     elif password != 'aawaaraa':
91.         return "Wrong Key"
92.     elif not save_csv(uid):
93.         return "ERROR: Invalid Arguments"
94.     return f"{uid}:{lat}:{long}"
95.
96.
97. if __name__ == '__main__':
98.     app.run(debug=True)
99.
```

Webpage HTML code-

```
1. <html>
2.   <head>
3.     <meta name="viewport" content="initial-scale=1.0, user-
   scalable=no">
4.     <meta charset="utf-8">
5.     <title>GPS</title>
6.     <link rel="stylesheet" href="../static/maa.css">
7.   </head>
8.   <body onload="initialize()">
9.     <div id="map_canvas" style="height: 500px"></div>
10.      <hr>
11.      <h1>Current Coordinates</h1>
12.      <h2>Latitude: {{lat}}</h2>
13.      <h2>Longitude: {{long}}</h2>
14.      <hr>
15.    </body>
16.    <script type="text/javascript"
   src="http://maps.googleapis.com/maps/api/js?libraries=geometry
   "></script>
17.    <script
   src="https://maps.googleapis.com/maps/api/js"></script>
18.    <script src="/static/where.js"></script>
19.    <script>
20.        function initialize() {
21.            var mapOptions = {
22.              zoom: 16,
23.              center: new google.maps.LatLng(last[0],last[1]),
24.              mapTypeId: google.maps.MapTypeId.ROADMAP
25.            }
26.            var map = new
   google.maps.Map(document.getElementById('map_canvas'),
   mapOptions);
27.            var i = 0;
28.            var markers = [];
29.          for(var point in data){
30.            var n = data[point]['data'].length;
31.            markers = [];
32.            for(i=1;i<n;i++){
33.              var line = new google.maps.Polyline({
34.                  path: [
35.                      new
   google.maps.LatLng(data[point]['data'][i-1][1],
   data[point]['data'][i-1][2]),
36.                      new
   google.maps.LatLng(data[point]['data'][i][1],
   data[point]['data'][i][2])
```

```
                ],
                strokeColor: data[point]['color'],
                strokeOpacity: 0.7,
                geodesic: true,
                strokeWeight: 3,
                map: map
              });
            }

            var marker = new google.maps.Marker({
                position: new
   google.maps.LatLng(data[point]['data'][n-1][1],
   data[point]['data'][n-1][2]),
                map: map,
                title: point
            });
            markers.push(marker);

          }
        }
    </script>
  </html>
```

Webpage CSS file-

```
body{
    color: white;
    margin: auto;
    width: 80%;
    background-color: black;
}
footer{
    text-align: center;
}
 .firebass{
     text-align: center;
 }
 .foota{
     text-decoration: none;
     color: blue;
 }
 .heee{
     text-align: center;
 }
 .hee{
     color: lightsalmon;
 }
 hr{
     background-color: red;
     border-width: 0px;
     height: 2px;
 }
 #sub{
     color: black;
 }
 a{
     text-decoration: none;
     color: white;
 }
 #main{
     width: 95%;
     margin: auto;
 }
 #main a{
     word-break: break-word;
     color: yellow;
 }
 * {
     box-sizing: border-box;
 }
 .container input[type=text], select, textarea {
```

```
    width: 100%;
    padding: 12px;
    border: 1px solid #ccc;
    border-radius: 4px;
    resize: vertical;
}
label {
    padding: 12px 12px 12px 0;
    display: inline-block;
}
.container input[type=submit] {
    background-color: #4CAF50;
    color: white;
    padding: 12px 20px;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    float: right;
}
input[type=submit]:hover {
    background-color: #45a049;
}
.container {
    border-radius: 5px;
    background-color: #f2f2f2;
    padding: 20px;
    width: 90%;
    transform: translate(5%, 0%);
}
.col-25 {
    float: left;
    width: 25%;
    margin-top: 6px;
    color: black;
}
.col-75 {
    float: left;
    width: 75%;
    margin-top: 6px;
}
.row1:after {
    content: "";
    display: table;
    clear: both;
}
@media screen and (max-width: 600px) {
    .col-25, .col-75, input[type=submit] {
        width: 100%;
        margin-top: 0;
```

```
    }
}
.row{
    border-spacing: 1vw;
    width: 90%;
    transform: translate(5%, 0);
}
.row .fir{
    position: relative;
    box-sizing: border-box;
    width: 80%;
    padding: 1px 5px;
    margin-inline-end:1%;
    vertical-align: top;
}
.row .sec{
    width: 20%;
    background-color: red;
    text-align: center;
    cursor: pointer;
}
.row:after {
    content: "";
    display: table;
    clear: both;
}

```

GPS module dump(for analysis only)

```
[NMEA] $PMTK010,001*2E [GPS1]
[NMEA] $PMTK011,VSMGPS*12 [GPS1]
[NMEA] $GPRMC,080902.475,V,,,,,,040521,,,N*4A [GPS1]
[NMEA] $GPGGA,080902.475,,,,,0,10,,,M,,M,,*7C [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,,,,*3A [GPS1]
[NMEA] $GPRMC,080938.064,V,,,,,,040521,,,N*47 [GPS1]
[NMEA] $GPGGA,080938.064,,,,,0,10,,,M,,M,,*71 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,,,,*3A [GPS1]
[NMEA] $GPRMC,080939.308,V,,,,,,040521,,,N*4F [GPS1]
[NMEA] $GPGGA,080939.308,,,,,0,10,,,M,,M,,*79 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,,,,*3A [GPS1]
[NMEA] $GPRMC,080940.558,V,,,,,,040521,,,N*42 [GPS1]
[NMEA] $GPGGA,080940.558,,,,,0,10,,,M,,M,,*74 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,,,,*3A [GPS1]
[NMEA] $GPRMC,080941.809,V,,,,,,040521,,,N*4A [GPS1]
[NMEA] $GPGGA,080941.809,,,,,0,10,,,M,,M,,*7C [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,,,,*3A [GPS1]
[NMEA] $GPRMC,080943.056,V,,,,,,040521,,,N*4A [GPS1]
[NMEA] $GPGGA,080943.056,,,,,0,10,,,M,,M,,*7C [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,,,,*3A [GPS1]
[NMEA] $GPRMC,080944.306,V,,,,,,040521,,,N*4B [GPS1]
[NMEA] $GPGGA,080944.306,,,,,0,10,,,M,,M,,*7D [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,,,,*3A [GPS1]
[NMEA] $GPRMC,080945.560,V,,,,,,040521,,,N*4C [GPS1]
[NMEA] $GPGGA,080945.560,,,,,0,10,,,M,,M,,*7A [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,,,,*3A [GPS1]
[NMEA] $GPRMC,080946.811,V,,,,,,040521,,,N*44 [GPS1]
[NMEA] $GPGGA,080946.811,,,,,0,10,,,M,,M,,*72 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,,,,*3A [GPS1]
[NMEA] $GPRMC,080948.057,V,,,,,,040521,,,N*40 [GPS1]
[NMEA] $GPGGA,080948.057,,,,,0,10,,,M,,M,,*76 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,,,,*3A [GPS1]
[NMEA] $GPRMC,080949.305,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*66 [GPS1]
[NMEA] $GPGGA,080949.305,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*67 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,080950.561,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*6A [GPS1]
[NMEA] $GPGGA,080950.561,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*6B [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,080951.809,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*68 [GPS1]
[NMEA] $GPGGA,080951.809,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*69 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,080953.056,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*68 [GPS1]
[NMEA] $GPGGA,080953.056,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*69 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,080954.308,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*67 [GPS1]
[NMEA] $GPGGA,080954.308,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*66 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,080955.561,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*6F [GPS1]
[NMEA] $GPGGA,080955.561,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*6E [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,080956.811,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*66 [GPS1]
[NMEA] $GPGGA,080956.811,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*67 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,080958.060,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*66 [GPS1]
[NMEA] $GPGGA,080958.060,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*67 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,080959.309,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*6B [GPS1]
[NMEA] $GPGGA,080959.309,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*6A [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
```

```
[NMEA] $GPRMC,081000.561,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*67 [GPS1]
[NMEA] $GPGGA,081000.561,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*66 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,081001.811,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*6C [GPS1]
[NMEA] $GPGGA,081001.811,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*6D [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,081003.056,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*65 [GPS1]
[NMEA] $GPGGA,081003.056,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*64 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,081004.307,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*65 [GPS1]
[NMEA] $GPGGA,081004.307,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*64 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,081005.558,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*68 [GPS1]
[NMEA] $GPGGA,081005.558,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*69 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,081006.812,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*68 [GPS1]
[NMEA] $GPGGA,081006.812,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*69 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,081008.062,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*69 [GPS1]
[NMEA] $GPGGA,081008.062,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*68 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,081009.308,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*67 [GPS1]
[NMEA] $GPGGA,081009.308,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*66 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,081010.558,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*6C [GPS1]
[NMEA] $GPGGA,081010.558,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*6D [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,081011.808,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*65 [GPS1]
[NMEA] $GPGGA,081011.808,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*64 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,081013.058,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*6A [GPS1]
[NMEA] $GPGGA,081013.058,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*6B [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,081014.308,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*6B [GPS1]
[NMEA] $GPGGA,081014.308,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*6A [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,081015.559,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*68 [GPS1]
[NMEA] $GPGGA,081015.559,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*69 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,081016.809,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*63 [GPS1]
[NMEA] $GPGGA,081016.809,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*62 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,081018.058,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*61 [GPS1]
[NMEA] $GPGGA,081018.058,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*60 [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,081019.869,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*6A [GPS1]
[NMEA] $GPGGA,081019.869,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*6B [GPS1]
[NMEA] $GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,,,4.00,3.20,2.40*0B [GPS1]
[NMEA] $GPRMC,081021.123,A,3404.2675,N,05159.7569,E,000.0,000.0,040521,,,A*66 [GPS1]
[NMEA] $GPGGA,081021.123,3404.2675,N,05159.7569,E,1,10,4.00,100.0,M,50.0,M,,*67 [GPS1]
```

GSM module dump(for analysis only)

```
AT
OK
ATE0
OK
AT+CPIN?
READY
AT+CIPSHUT
OK
AT+CGATT=1
OK
AT+CSTT="airtelgprs.com","",""
OK
AT+CIICR
OK
AT+CIPSTART="TCP","127.0.0.1",5000
CONNECT
AT+CIPSEND=69
GET /add?lat=3404.2675N&long=05159.7569E&key=aawaaraa&id=1 HTTP/1.0


AT+CIPSTART="TCP","127.0.0.1",5000
CONNECT
AT+CIPSEND=69
GET /add?lat=3404.2675N&long=05159.7569E&key=aawaaraa&id=1 HTTP/1.0
```

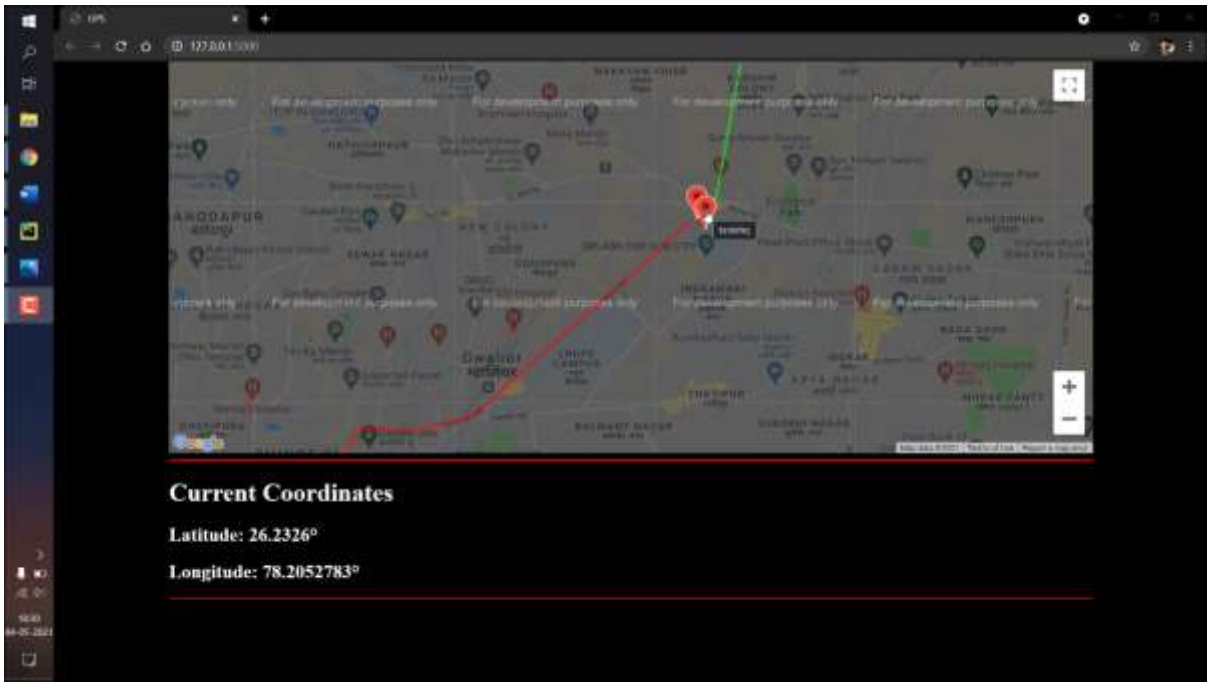
PROCEDURE

1. Install the Proteus design suite.
2. Instal the Arduino IDE.
3. Install Python and then Flask
4. Type the code you want to upload in the Arduino IDE.
5. Now go to Sketch -> Export Compiled Binary, it will generate the required HEX file.
6. Now open Proteus and make new project.
7. Go to pick components.
8. Select the required components.
9. Select resistor, capacitor, ATmega328p, GPS module, LED, Virtual terminal and other if required.
10. Place the components on the screen.
11. Edit the values of the components as per the requirements.
12. Connect the components according to the schematic diagram given above.
13. Now open the properties of the MCU unit, in program click the browse icon and the select the HEX file without bootloader generated in step 4.
14. Now run the simulation.
15. Now while running reply to the AT commands as they are appearing on virtual terminal to imitate the GSM module.
16. Type the code for Flask server.
17. Run the flask server by running the python file
18. Now open the links as you got them from MCU.
19. Now visit the homepage and see the result.

REAL OUTCOME

WEBAPGE

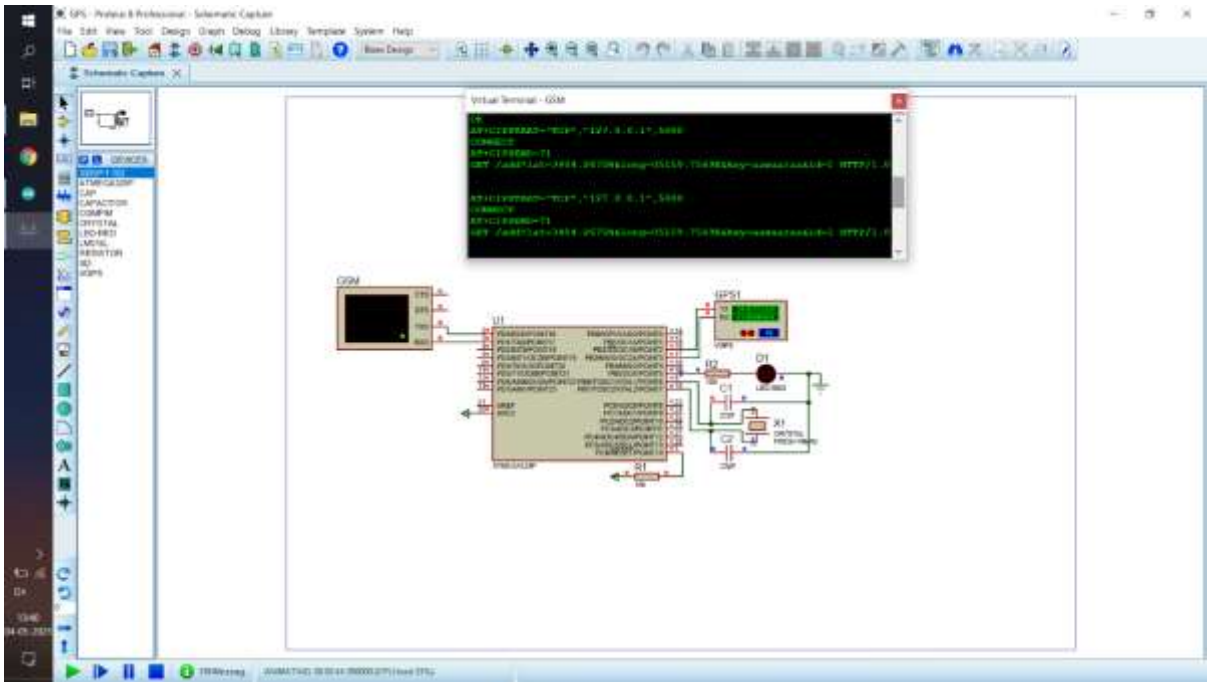
Photo of website showing the path taken by two users and their last updated location. Two different user's paths are shown by different colour, red and green. Also hovering over the marker shows the name of the user present at that location.



In the above picture we can also see the Current coordinates which are the last updated coordinates from the last moving user.

HARDWARE

In this picture we can see the simulation running in proteus, where GPS module is transmitting the coordinates to the microcontroller and then the GSM module here is simulated by the use of virtual terminal, where the AT commands are transmitted from microcontroller and their reply is typed on virtual terminal, to imitate the GSM module working, here we can see how the AT commands from data transfer are being sent continuously.



CONCLUSION

Vehicle tracking system are getting popular day by day not only in metropolitan areas but also in small cities. This system is completely integrated and it becomes possible for the user to track their car very easily at any time and from anywhere. As the vehicle theft is increasing day by day but due to this people will avoid using vehicles but they found an efficient way to keep an eye on their vehicle without being very close to them. These systems keep a good control on the thefts and help avoiding them to some extent. Basically, in all these systems the GPS and GSM are used to track the vehicle. Using this system, the user determines the position of the vehicle, and the distance completed by it. The user is able to access the position of their vehicle at any instant of time. This system is reliable any very secure. Upgrading this setup is very easy which makes it open to future requirements without the need of rebuilding everything from scratch, which also makes it more efficient.

Apart from safety from theft this technology also provides a move faster in places. Suppose the ambulance went to take a patient, so when the ambulance will return we know in advance of 100m(or something) then we can necessary arrangements for the arrival of the patient beforehand, without wasting manpower by making one person stand there to tell the if the ambulance came or not. We can also use this technology on toll plazas, then we even don't need to setup a building for toll plaza, we can just set up the virtual boundary and anyone who crosses this pays the toll, it will save lot of manpower and resources.

We can track if your child/employee is breaking the rule or going outside of certain boundary or he/she is not doing the work assigned and going to some other place without notifying.

We can also use this for spying, when connected to connected and using some glue, we can stick this to someone's vehicle without their knowledge and gain access to their location.

FUTURE ASPECTS

•Free-floating car-sharing

A special utility program enables the user to find, select, and reserve vehicles in the city region, depending on where the customer is at the moment. The smartphone can navigate you to the car's location and even open the car for you: Digital car keys integrated into a conventional smartphone are already revolutionizing the spontaneous hiring of rental cars. In the future, GPS-based services could enable autonomous car-sharing and parking, with vehicles that pick you up where you are and leave you at your destination to park themselves – or to pick up the next customer.

•Automated payment systems

When you for example enter and leave a car-sharing vehicle or the subway, a location-based service computes your fare automatically and adds the amount to your monthly fee or debits the sum from your bank account right away.

•Real time alert-

A tracking system can be an excellent investment to improve customer satisfaction levels in your delivery operations. This is because, with a system like that in place, managers can quickly set alerts to notify customers when their packages enter their POI zone and notify an estimated time of arrival.

- Geo-fencing

It is a feature in a software program that uses the global positioning system (GPS) or radio frequency identification (RFID) to define geographical boundaries. Geo-fencing allows an administrator to set up triggers when a device or gadget enters (or exits) the boundaries defined or characterized by the administrator, an alert is issued. Many geo-

fencing applications incorporate Google Earth, allowing administrators to define boundaries on top of a satellite view of a specific geographical area. Other applications define boundaries by longitude and latitude or through user-created and Web-based maps. Whereas Geofencing provides the ability to mark those points as POI (Points of interests), monitor the vehicle's movements around these points of interest, and communicate in real-time about the activities and interests of the consumers.

REFRENCES

- https://m2msupport.net/m2msupport/tcpudp-ip-testing-for-m2m-modules/
- https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf
- https://www.sparkfun.com/datasheets/GPS/Modules/LS20030~3_datasheet_v1.2.pdf
- https://flask.palletsprojects.com/en/1.1.x/quickstart/#a-minimal-application
- https://developers.google.com/maps/documentation/javascript/overview