VIT®
UNIVERSITY
(Estd. u/s 3 of UGC Act 1956)
VELLORE ■ CHENNAI
www.vit.ac.in

SMART PARKING SYSTEM

Supervised By- KRITHIKA L.B

(Object Oriented Analysis and Design)

Submitted by: -

16BIT0177 -	ANKIT KUMAR
16BIT0044 -	PRANSHU PRANJAL

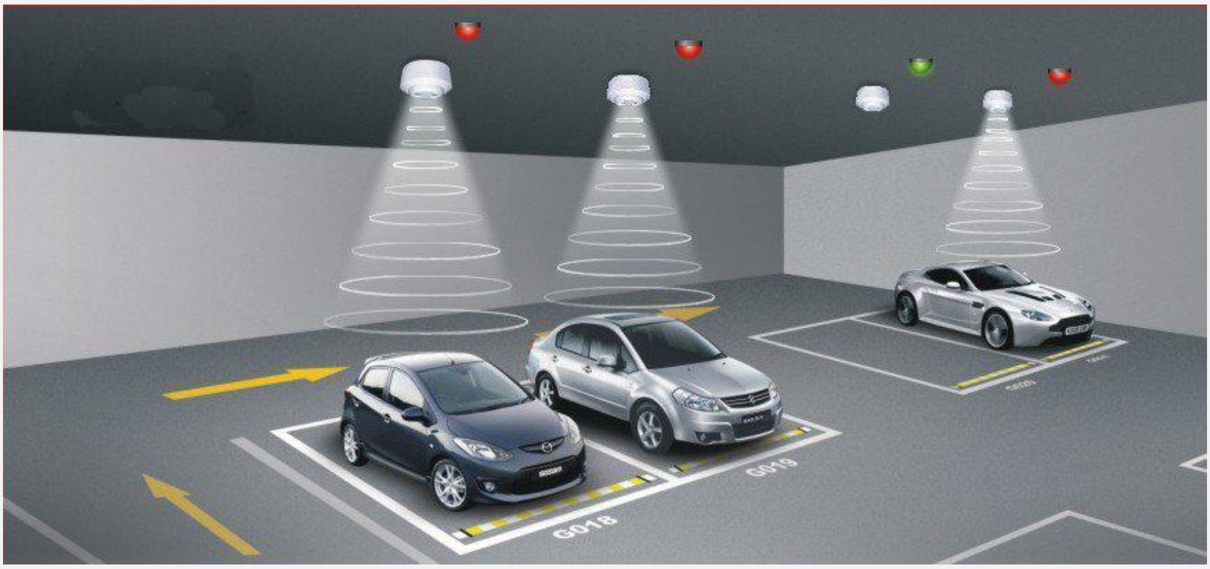
Team Number: 8

TITLE:-- Smart Online Car Parking System

Abstract

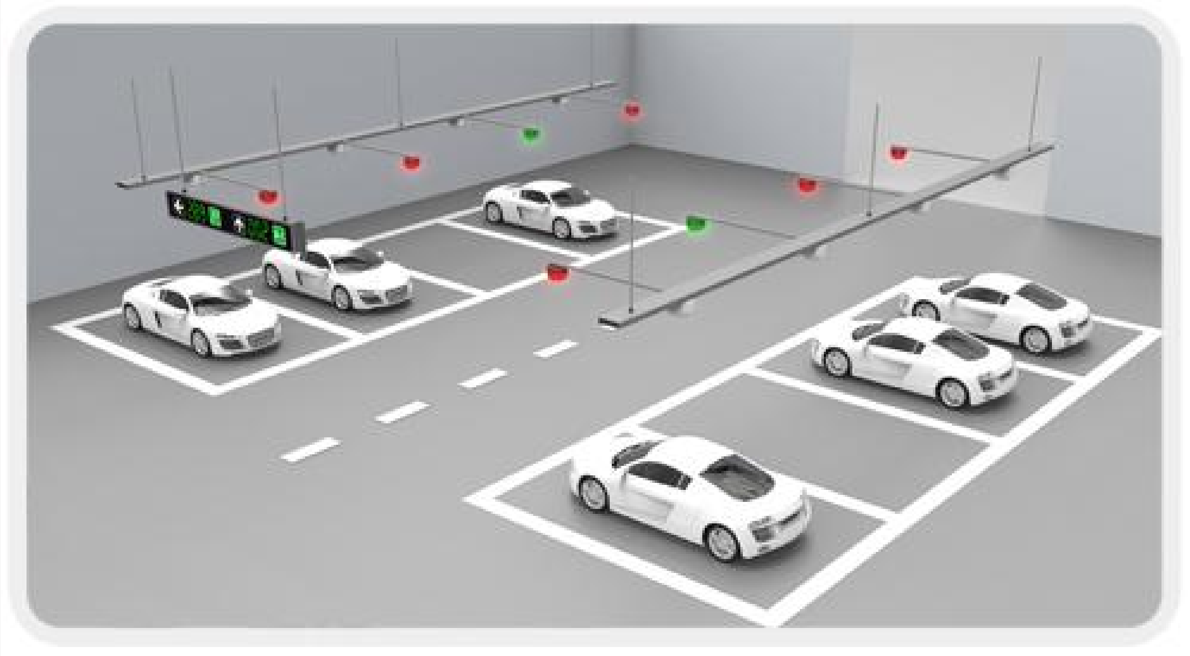
Due to the proliferation in the number of vehicles on the road, traffic problems are bound to exist. This is due to the fact that the current transportation infrastructure and car park facility developed are unable to cope with the influx of vehicles on the road. To alleviate the aforementioned problems, the smart parking system has been developed. The implementation of the smart parking system is made possible using Google Map. We can easily locate and secure a nearby vacant parking space at any car park deemed convenient to us as a customer.

In constructing the parking system for a project activity, the proposed design was done using HTML, CSS, JAVASCRIPT and the GOOGLE API have been done using SUBLIME TEXT EDITOR.



INTRODUCTION

Smart Parking System is a vehicle parking system that helps drivers find a vacant spot. Using sensors in each parking space that detect the presence or absence of a vehicle, signs direct incoming drivers to available locations. Smart parking can be defined as the use of advanced technologies for the efficient operation, monitoring, and management of parking within an urban mobility strategy. The global market for smart parking systems reached $93.5 million, with the United States representing 46% market share, and offering a strong growth opportunity for companies offering services in the United States and overseas. A number of technologies provide the basis for smart parking solutions, including vehicle sensors, wireless communications, and data analytic. Smart parking is also made viable by innovation in areas such as smart phone apps for customer services, mobile payments, and in-car navigation systems. At the heart of the smart parking concept is the ability to access, collect, analyze, disseminate, and act on information on parking usage. Increasingly, this information is provided in real-time from intelligent devices that enable both parking managers and drivers to optimize the use of parking capacity.



EXISTING SYSTEM

- **GPS based systems**

The information about the location and availability of a parking space near the destination is provided to the drivers by the current GPS-based vehicle navigation system. The information of the current state of the parking facility is provided. That's why they can't guarantee a parking lot when the driver reaches the facility. A scientific solution based on utilization of the past and current status of the parking lot is proposed. Poisson process is used for modeling the availability of a parking lot. An intelligent algorithm which helps the driver in choosing the slot with maximum probability of being vacant is presented as well. Cerreo had a yearlong study on street parking, which focused on different issues concerned with on-street parking namely policy, planning, management and operations. Various methods and different cities were used for demonstrating these issues. It also highlighted different challenges of on-street parking such as peer-to-peer exchange and storage of parking information.

- **Wireless sensor based systems**

A number of low cost sensor nodes make up a Wireless Sensor Network (WSN). They arrange themselves for making an ad hoc network through the wireless communication module present on nodes. Different types of sensors, computation units and storage devices are present on each node. For collecting, processing and transmitting information, the functional parts let the sensors to be installed quickly and easily. They have a bright future since WSNs are easily installed in the normal environment and they provide the data for positioning and surveillance. There are two disadvantages associated with the video sensors. The first one is that video sensors are expensive. Secondly, sensors generate huge data at times; whose transmission via the Wireless Network is hectic.

- **Other miscellaneous techniques**

A parking reservation service for reserving a vacant parking spot via the internet is proposed . This system utilizes the internet for a quick and easy search for the vacant lot. By using it together with a smart card, the system also provides recognition and payments services to the driver. a multilevel driver assistance system for assisting in the parking process is proposed. A parking assistance system along with parking administration system, and employed sensor systems are mentioned as well. A general architecture of a driver assistance system which relies on path planning and Human-Machine interface (HMI) modules is presented as well.

What are the problem with existing smart parking?

All of the smart parking solutions are available only in big areas like mall, metro parking or in big cities' famous landmarks areas. They not provide solutions for local areas in day to day life. Like if we go to a nearby market and need to find a place to park our car, we have to search manually by driving by all the known parking locations and still we may not be able to find appropriate space.

Proposed System

Our smart parking technology **optimized parking** *(using appropriate spaces)*, **reduced traffic** *(preventing parking at roads)*, **increased safety** *(by preventing haphazard parking of vehicles)*, **maximum efficiency** *(using all parking space)* for anywhere like on street or any local area or building.

- ✓ If a person goes to a new place and needs a place to park his/her vehicle, then we would be guiding him to the nearest parking place.

- ✓ We would also be offering the service of finding your nearby parked vehicle if you get lost or can't reach it yourself.

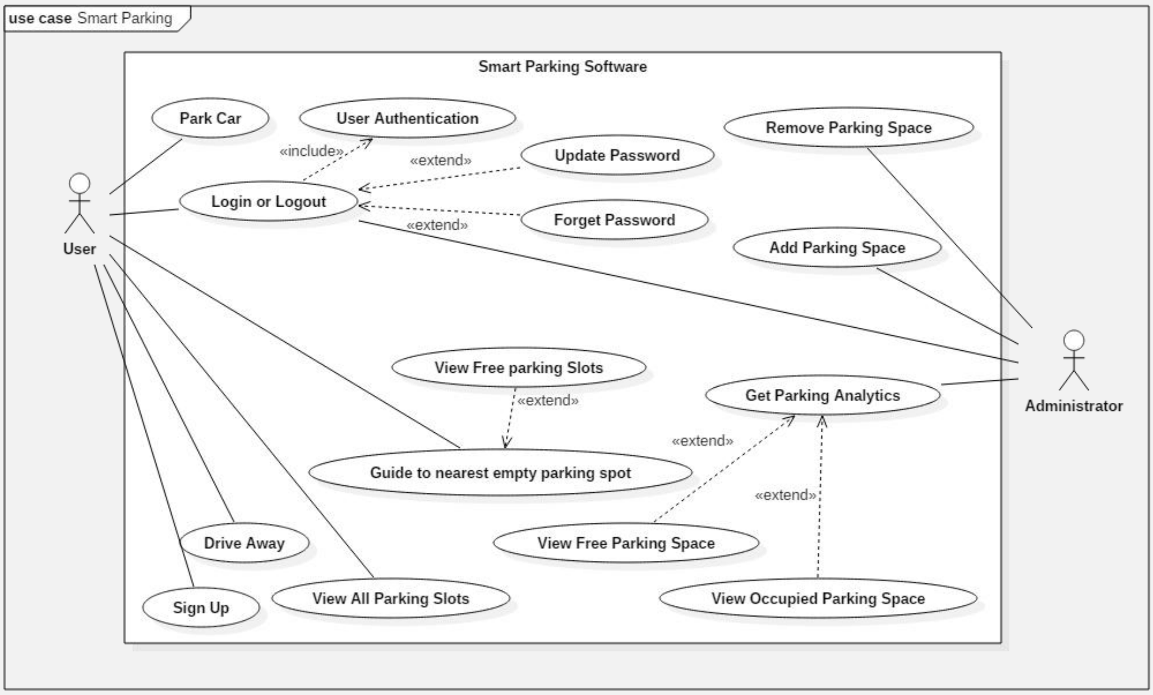
- ✓ Also we are planning to integrate the feature (if possible) like if your friend's car is parked nearby it will give a prompt that your friend could be nearby, why don't you meet him/her.

Guiding to the correct location will be based on the current location and the parking location given in our database based on Google Map.

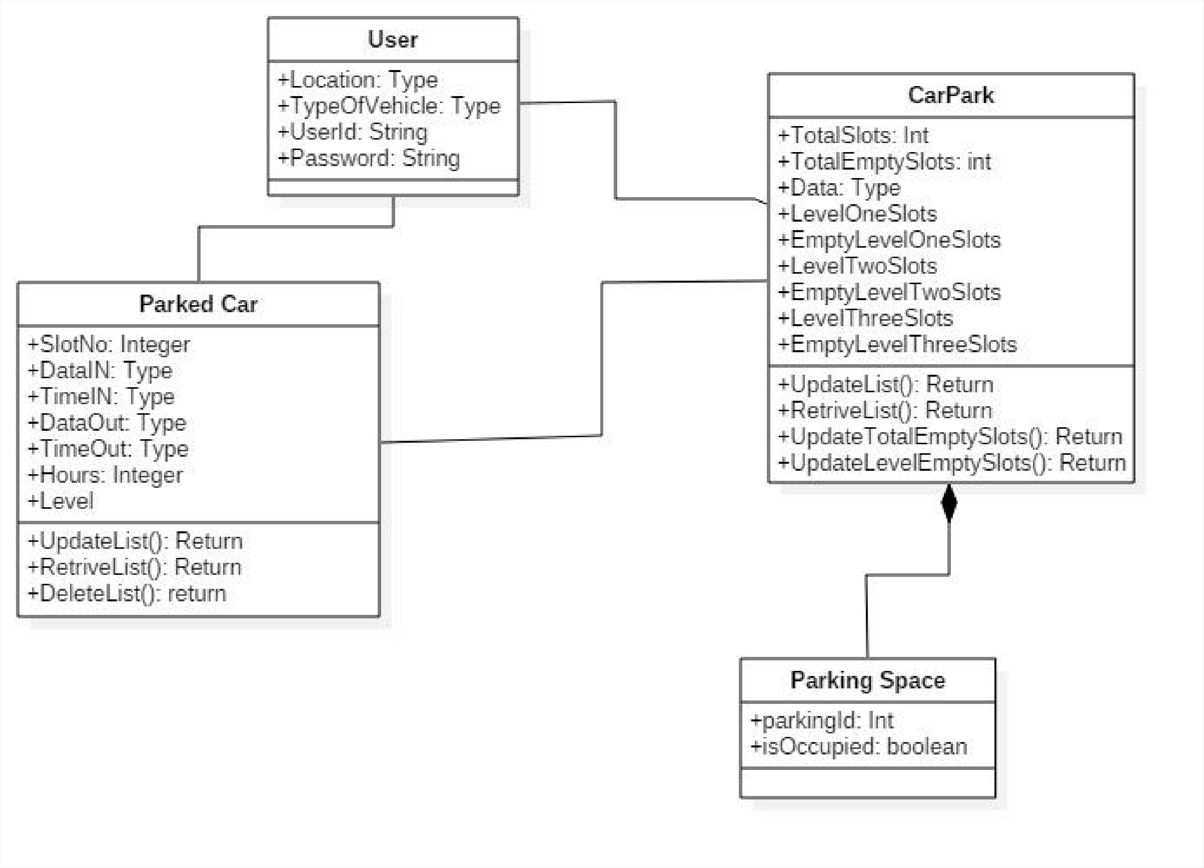
We can't actually implement using sensors so would be making the project based on some **pre-loaded** parking destinations.

UML DIAGRAMS

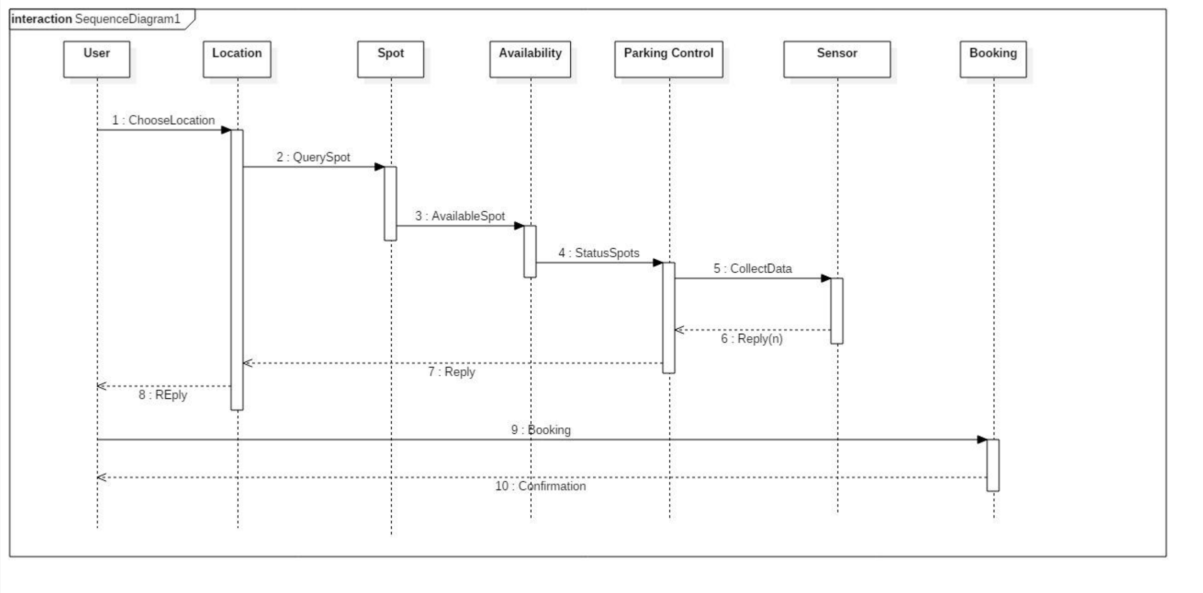
Use Case Diagram



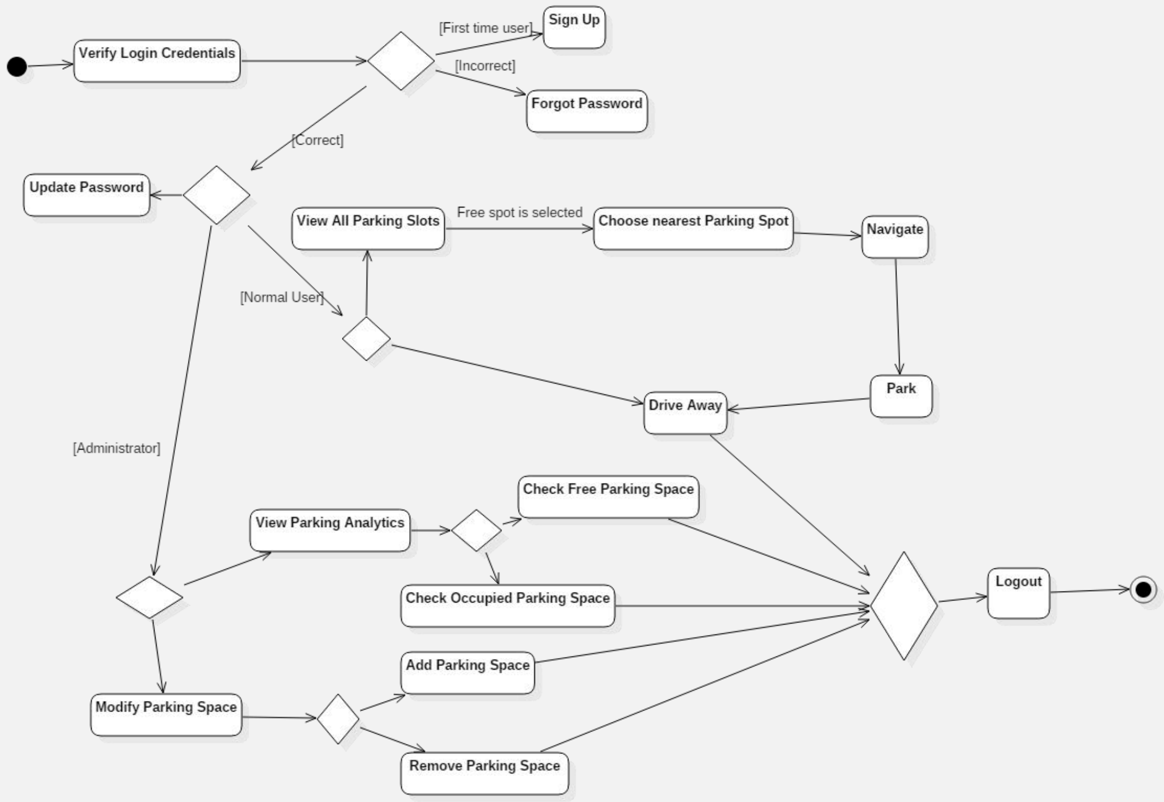
Class Diagram



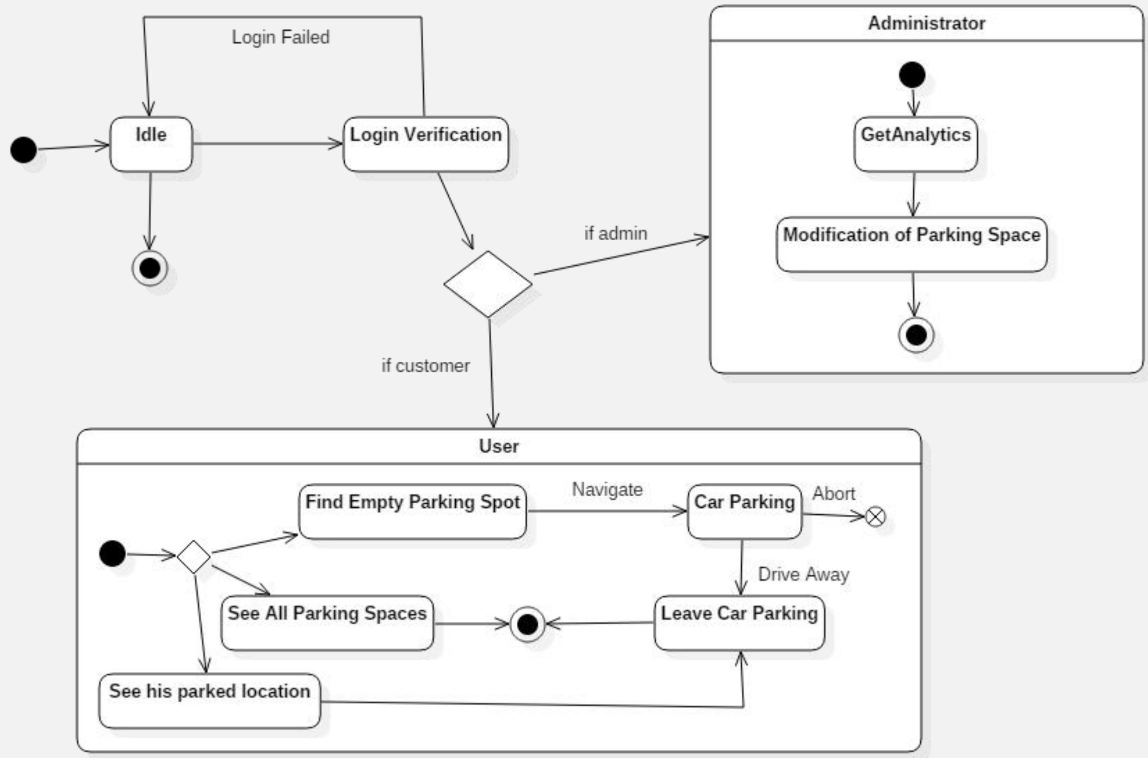
Sequence Diagram



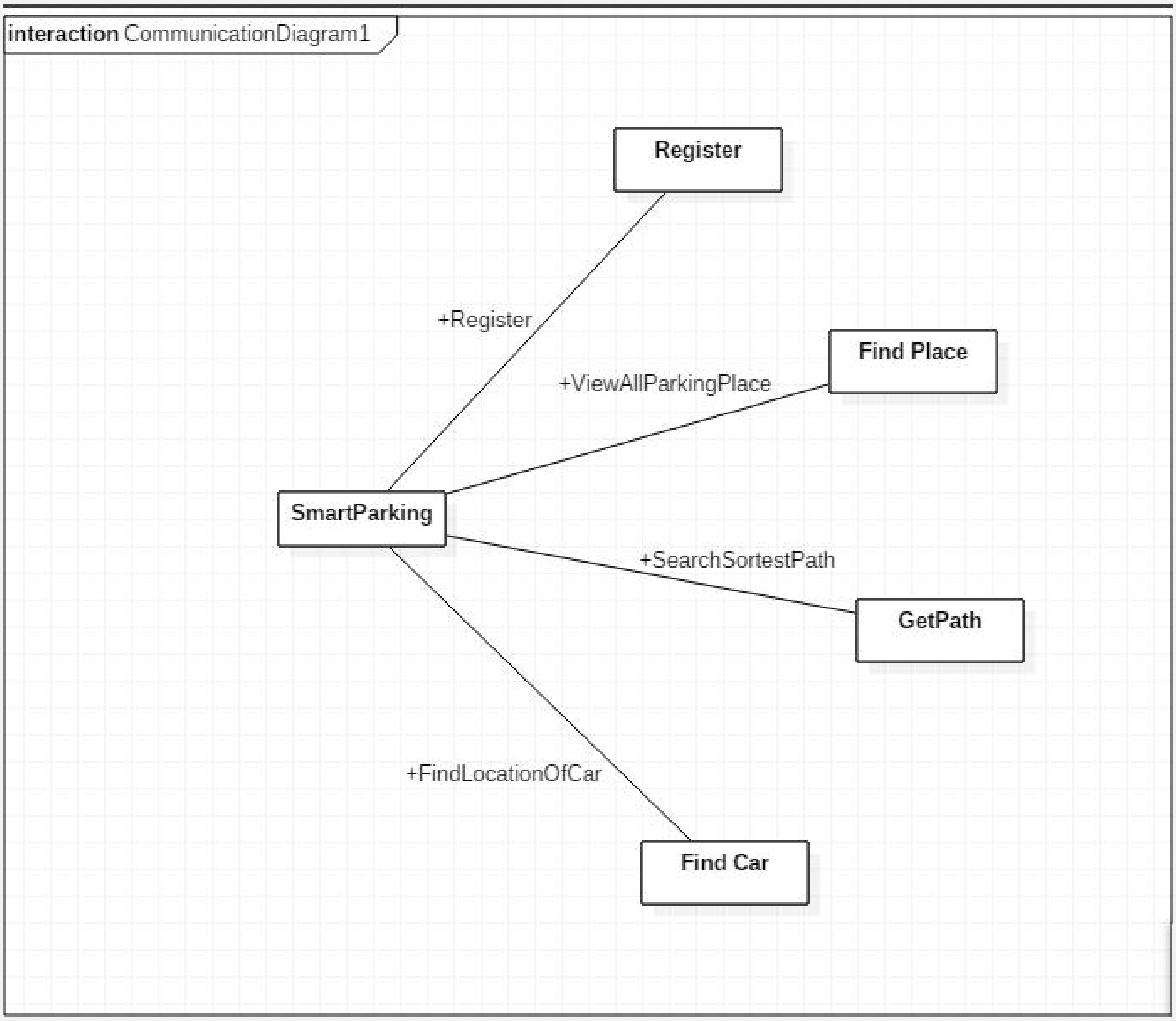
Activity Diagram



State Transition Diagram



Collaboration Diagram



IMPLEMENTATION

The Project is mainly based on Google Maps API (Free Version, 25000 hits per day)

We Use Haversine Formula For Findind Distance.

Explanation--

The haversine formula is an equation important in navigation, giving great-circle distances between two points on a sphere from their longitudes and latitudes. It is a special case of a more general formula in spherical trigonometry, the law of haversines, relating the sides and angles of spherical "triangles".

JAVASCRIPT CODE:--

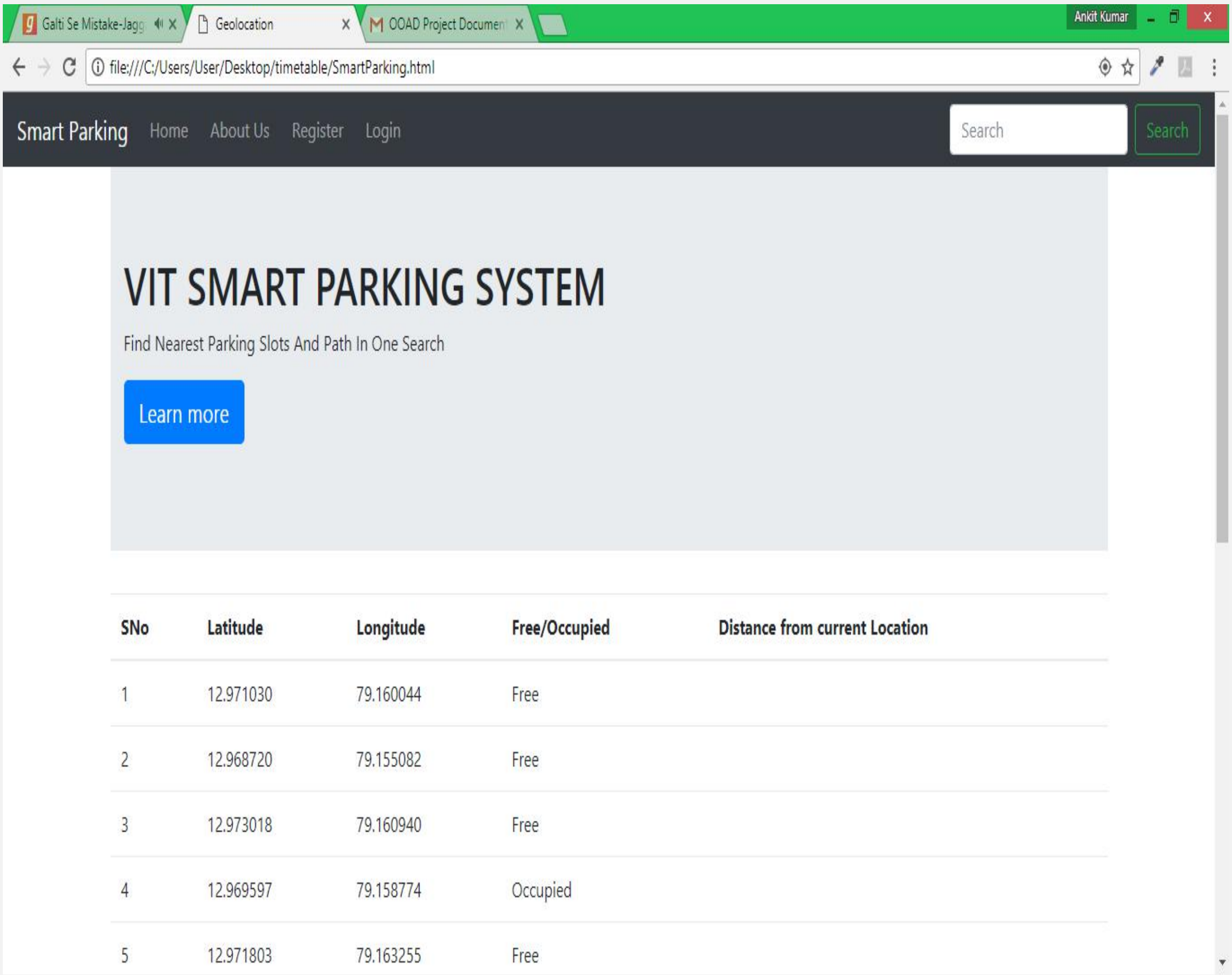
```
<script>
    var x=12.971803;
    var y=79.163255;

    var map, infoWindow;

          var uluru,pos;

    function initMap()
    {
                var directionsService = new google.maps.DirectionsService();
                var directionsDisplay = new google.maps.DirectionsRenderer();

        uluru = {lat: x, lng: y};

     map = new google.maps.Map(document.getElementById('map'), {
      center: {lat: 12.971762, lng: 79.159989},
  zoom: 18
});

directionsDisplay.setMap(map);

var marker = new google.maps.Marker({
  position: uluru,
  map: map,
  icon: 'http://maps.google.com/mapfiles/ms/icons/blue-dot.png'
});


if (navigator.geolocation)
{
  navigator.geolocation.getCurrentPosition(function(position)
  {
    pos = {
      lat: position.coords.latitude,
      lng: position.coords.longitude
    };


    var marker1 = new google.maps.Marker({
      position: pos,
          map: map
            });

    map.setCenter(pos);

  }, function() {
      console.log('Pranshu Prajal')

    });

  }

  var onChangeHandler = function() {
    calculateAndDisplayRoute(directionsService, directionsDisplay);
  };
  document.getElementById('traceit').addEventListener('click', onChangeHandler);
}

function calculateAndDisplayRoute(directionsService, directionsDisplay) {
  directionsService.route({
    origin: pos,
    destination: uluru,
    travelMode: 'DRIVING'
  }, function(response, status) {
    if (status === 'OK') {
      directionsDisplay.setDirections(response);
    } else {
      window.alert('Directions request failed due to ' + status);
    }
  });
 }


 function extractcoordinates(pos,str_lat,str_lng){
  pos.lat;
```

```
   console.log(pos.lat);
   pos.lng;
   document.getElementByClassName('lat');
  }


 function constructDist() {
    var table=document.getElementById('database');

   for(var i=1; i<table.rows.length;i++)
   {
     var row = 0;
     var str_lat=(table.rows[i].cells[1].innerHTML);
     var str_lng=(table.rows[i].cells[2].innerHTML);
         var distance=getDistanceFromLatLonInKm(str_lat,str_lng,pos.lat,pos.lng);
         table.rows[i].cells[4].innerHTML=distance;
   }


var smallest=table.rows[1].cells[4].innerHTML;
console.log(smallest);

for(var xq=2; xq<table.rows.length;xq++)
{
  if(table.rows[xq].cells[3].innerHTML=='Free')
     {

       if(parseFloat(smallest)>parseFloat(table.rows[xq].cells[4].innerHTML))
       {
         smallest=table.rows[xq].cells[4].innerHTML;
```

```
            }
            else
            {
              console.log("error");
            }
          }
     }

      for(var m=1; m<table.rows.length;m++)
      {
         if(table.rows[m].cells[4].innerHTML==smallest)
         {
            x=parseFloat(table.rows[m].cells[1].innerHTML);
            y=parseFloat(table.rows[m].cells[2].innerHTML);
            uluru = {lat: x, lng: y};
         }
      }

}




          function getDistanceFromLatLonInKm(lat1,lon1,lat2,lon2) {
            var R = 6371; // Radius of the earth in km
            var dLat = deg2rad(lat2-lat1);  // deg2rad below
            var dLon = deg2rad(lon2-lon1);
            var a =
              Math.sin(dLat/2) * Math.sin(dLat/2) +
              Math.cos(deg2rad(lat1)) * Math.cos(deg2rad(lat2)) *
              Math.sin(dLon/2) * Math.sin(dLon/2)
```

```
    ;
   var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a));
   var d = R * c; // Distance in km
   return d;
  }

  function deg2rad(deg) {
   return deg * (Math.PI/180)
  }
</script>
```

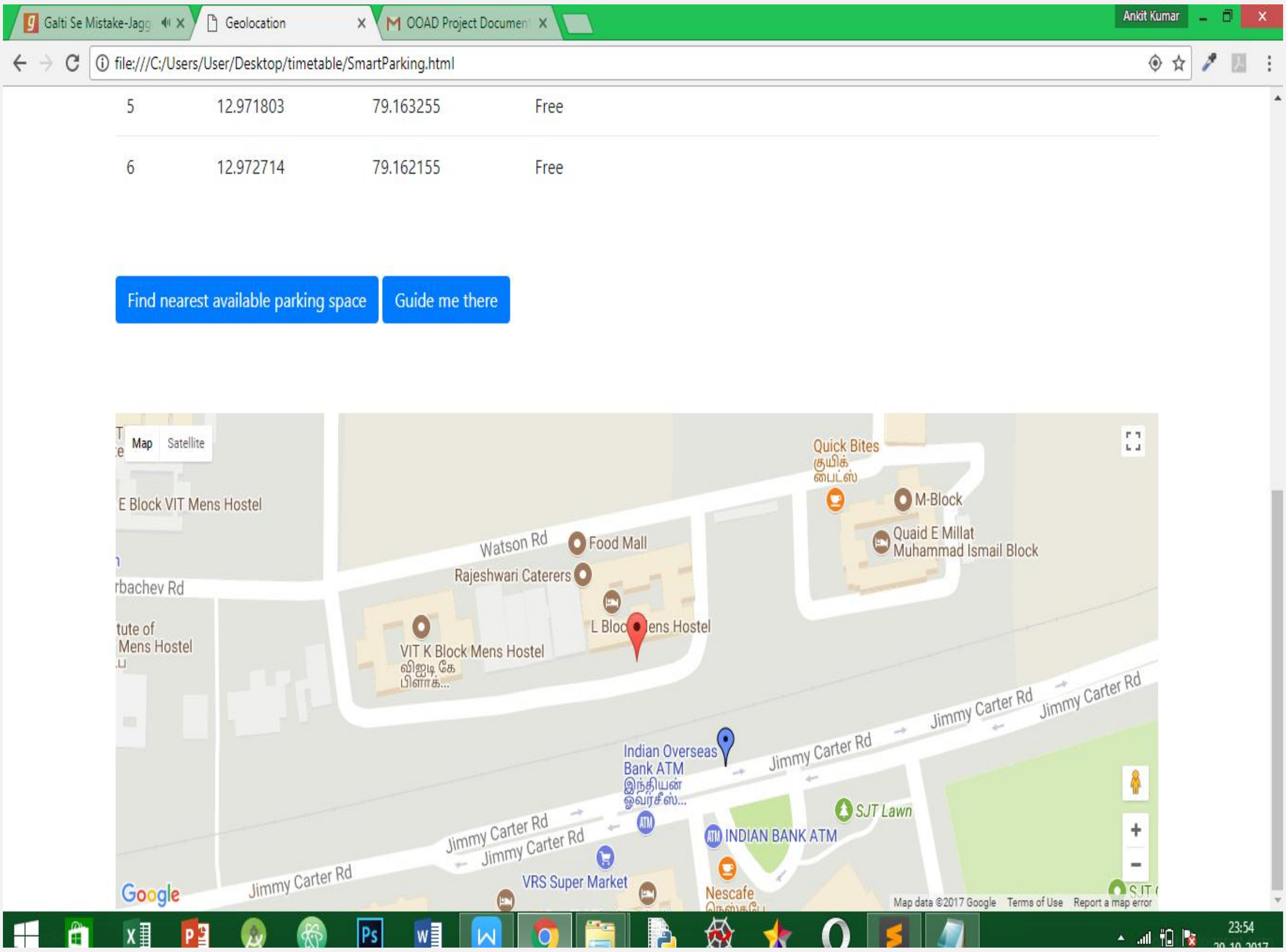
USER INTERFACE--



VIT SMART PARKING SYSTEM

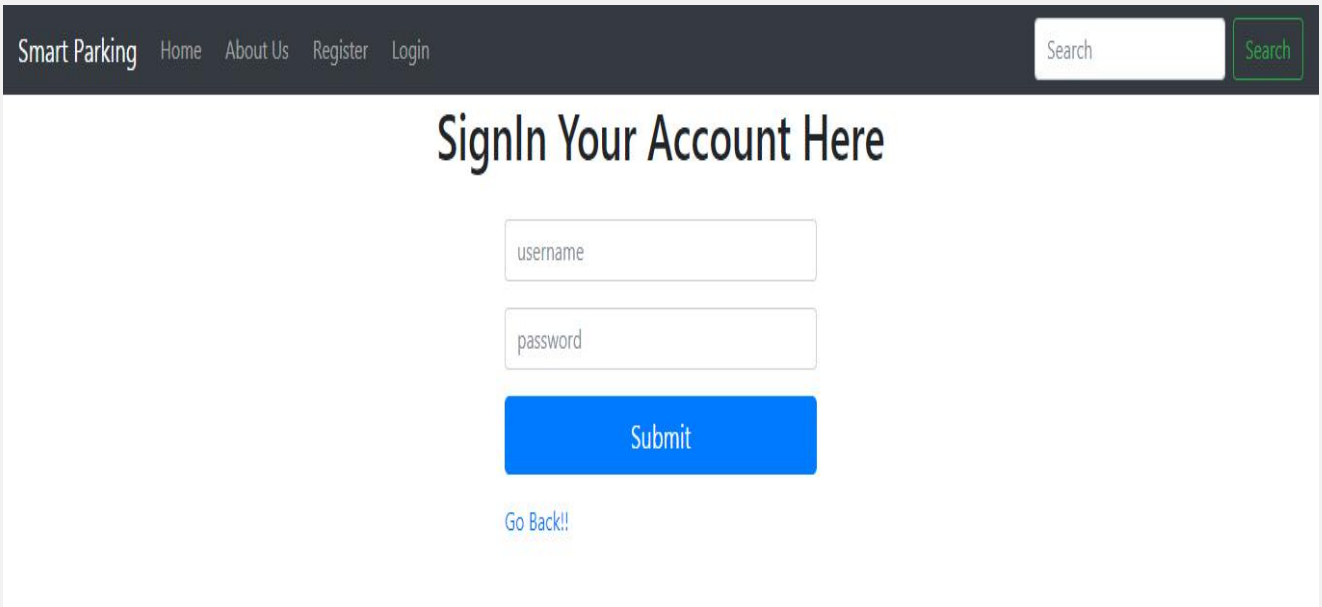
Find Nearest Parking Slots And Path In One Search

SNo	Latitude	Longitude	Free/Occupied	Distance from current Location
1	12.971030	79.160044	Free	
2	12.968720	79.155082	Free	
3	12.973018	79.160940	Free	
4	12.969597	79.158774	Occupied	
5	12.971803	79.163255	Free	



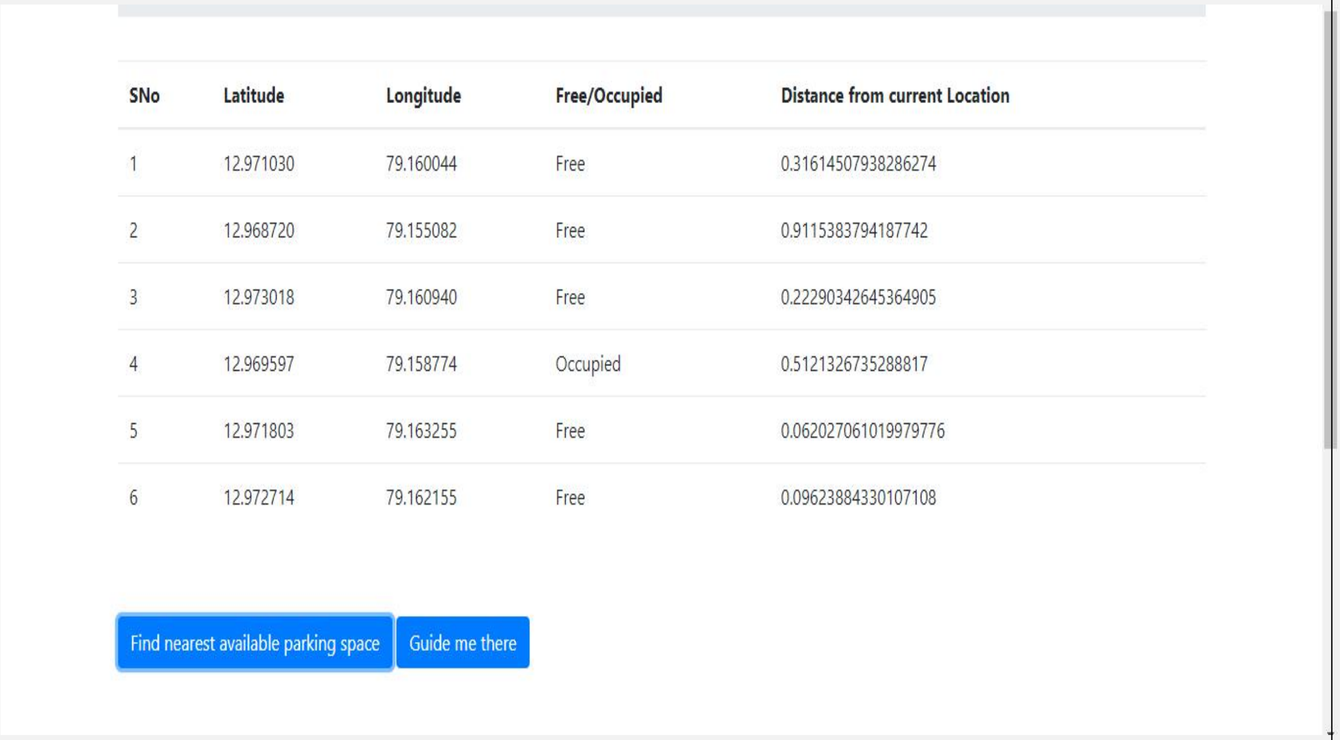
5	12.971803	79.163255	Free
6	12.972714	79.162155	Free

LOGIN --



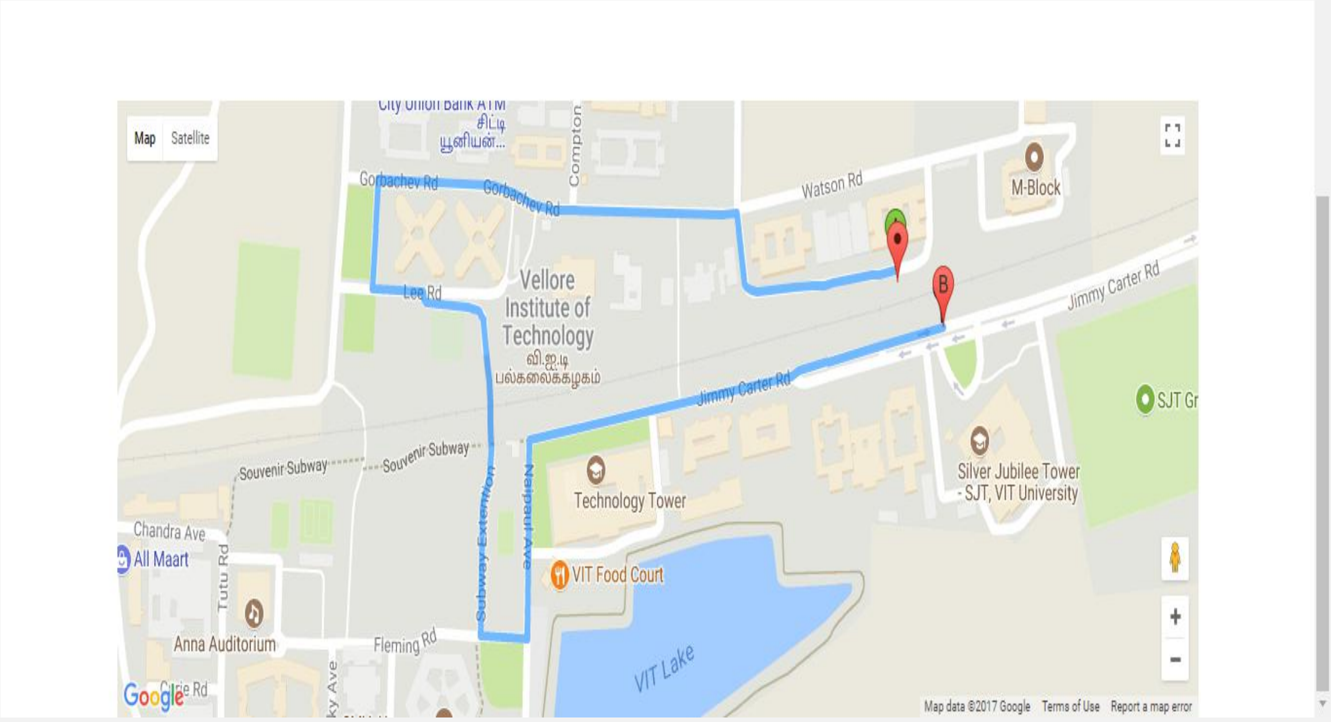
SignIn Your Account Here

ALL THE NEAREST PARKING SLOTS DISTANCE FROM YOUR CURRENT LOCATIO --



SNo	Latitude	Longitude	Free/Occupied	Distance from current Location
1	12.971030	79.160044	Free	0.31614507938286274
2	12.968720	79.155082	Free	0.9115383794187742
3	12.973018	79.160940	Free	0.22290342645364905
4	12.969597	79.158774	Occupied	0.5121326735288817
5	12.971803	79.163255	Free	0.062027061019979776
6	12.972714	79.162155	Free	0.09623884330107108

GUIDE THE WAY OF NEAREST PARKING SLOT --



CONCLUSION

The Project was made to tackle the problem of finding parking space. It was made using Google Maps API to find the nearest empty parking space and find a route to it. The Project was made using HTML, CSS, JavaScript under the guidance of Prof. Krithika L. B. The made web app helps to find the nearest available parking location based on Haversine Formula and would be great help for all the clients. The current location and pre-defined park locations are used to find the shortest distance and then show the most direct route for the users to get to the nearest available parking location in the minimum amount of time.