

Public Transport Optimization using IoT with Sensors

Project Description:

This project aims to solve the problem of tracking and accountability of vehicles by providing a software platform. This project would serve as an important step to help in Vehicle tracking, component monitoring, vehicle analysis and fleet management. An efficient vehicle tracking system is implemented for monitoring of any equipped vehicle from any location at any time with the help of Global Positioning System (GPS) and Arduino Board which will enable users to locate their vehicles with ease and in a convenient manner.

GPS Sensor:

Application: Used for real-time vehicle tracking and route optimization.

Benefits: Provides accurate location data, helping to monitor vehicle movement, calculate ETA, and optimize routes based on traffic conditions.

Application: This project can be deployed in various fields like elderly and disabled care services, logistic division, emergency services and rescue operation, school bus tracking, and accounting...

1. Real-Time Tracking

Here, the GPS receiver receives the location data like latitude and longitude of a vehicle and sends them by using an HTTP request to web server. The browser is used to load the PHP web page which contains Google maps to show the location of the vehicle in real time. The web page containing the map directly marks the coordinates, as it arrives, without reloading the page. That means, in real time, we get to see the location of the vehicle.

2. History

In history, we ask the user to select the date of journey, the names of vehicles which were on a journey on that day are displayed by selecting the vehicle the journey of that vehicle on that date is fetch. The google map with a marker is displayed on the screen. Along with the map, the Time-Speed Graph of journey and details of the journey is displayed.

In history, the user must select the date of journey and correspondingly the names of vehicles active on that day are displayed. After the vehicle has been selected, corresponding journey details are displayed. Journey details include google maps representing journey, graph showing speed of vehicle at every instance of time and driver details consisting of driver details.

3. Report Generation

This part includes generation of specific reports for drivers, trucks, and journeys. In this

pdf report is generated, which are used by owners for analysis. For the driver's report, driver must be selected which then generates a report showing basic details and journeys done by him. For truck report, truck must be selected which then generates a report showing basic details and journeys done by him. For journey report, date and vehicle need to be selected which then generates a report showing basic journey details, google maps representing complete journey and graph showing speed of vehicle at every instance of time.

4. Geofencing

Geofencing is one of the important features of our project. As the problem statement stated that driver may take a longer route to increase fuel consumption and ask for more money for his long journey or the driver unintentionally took the wrong path, such scenario affects the throughput of the journey. It also affects the performance of vehicles and drivers. Thus, geofencing is applied.

Geofencing is the virtual boundary to the optimized path from source to destination which is defined at the time of journey creation. It is a radius which the user has to input in meters. Whenever vehicle deviates from its optimized path and the deviation is more than the defined radius then it means it's a geofencing break, which results in a notification popup generated on the screen. This helps the user to know that geofencing is broken by this vehicle.

5. SMS Module

SMS module is an important feature for the user. As the project is for vendors and they are very busy they can't stay in front of the computer and check the current location of the vehicle. In this scenario, the SMS module is useful to know the current location of vehicle. To know the current location of the vehicle user, you must send an SMS to several of the sims attached to the hardware. In response, the SMS a link is generated when user clicks on that link on the browser it shows a google map with the marker. The marker is the current location of the vehicle at that time. Thus, the SMS module helps the user to check the location of vehicle anytime he wants.

6. Agile Hardware

Hardware is reliable, low cost and compact. As we stated that this project has various applications in various areas so for those applications' hardware will be fixed only, we must change the web application and the method to use the functionality. The functionality and construct of this hardware is fixed and working for this.

Languages: C, PHP, JavaScript, Ajax, Bootstrap, HTML, CSS

Hardware: Arduino UNO, SIM808, GPS antenna

DEVELOPING THE REAL-TIME TRANSIT INFORMATION PLATFORM BY USING WEB TECHNOLOGY

```
<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Real-Time Transit Information</title>

  <link rel="stylesheet" href="styles.css">

</head>

<body>

  <header>

    <h1>Real-Time Transit Information</h1>

  </header>

  <main>

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

    <div id="schedule">

      <h2>Upcoming Buses</h2>

      <ul id="bus-schedule">

        <!-- Real-time bus schedule data will be displayed here -->

      </ul>

    </div>

  </main>

  <script src="script.js"></script>

</body>

</html>
```

CSS

```
body {

  font-family: Arial, sans-serif;

  margin: 0;

  padding: 0;

}
```

Header

```
{  
  background-color: #333;  
  color: #fff;  
  text-align: center;  
  padding: 20px;  
}
```

Main

```
{  
  display: flex;  
  justify-content: space-between;  
  padding: 20px;  
}
```

```
{  
  flex: 1;  
  height: 400px;  
}
```

```
{  
  flex: 1;  
  padding: 10px;  
}
```

```
#bus-schedule {  
  list-style-type: none;  
  padding: 0;  
}  
  
{  
  margin: 5px 0;  
  border: 1px solid #ccc;  
  padding: 10px;  
}
```

Java script

```
document.addEventListener("DOMContentLoaded", () => {
  const busSchedule = [
    { route: "Bus 101", destination: "Downtown", nextArrival: "10:15 AM" },
    { route: "Bus 202", destination: "Suburbia", nextArrival: "10:30 AM" },
    { route: "Bus 303", destination: "Airport", nextArrival: "10:45 AM" },
  ];
  const busScheduleList = document.getElementById("bus-schedule");
  busSchedule.forEach((bus) => {
    const listItem = document.createElement("li");
    listItem.innerHTML = `<strong>${bus.route}</strong> to ${bus.destination}<br>Next arrival:
    ${bus.nextArrival}`;
    busScheduleList.appendChild(listItem);
  });
});
```

DESIGN THE PLATFORM TO RECEIVE AND DISPLAY REAL-TIME DATA FROM IOT SENSORS:

