Public Transport Optimization

Using IOT

Abstract:

The final part of the project



By,

Deepak V - au721421106011

Hariharan G - au721421106021

Esakki Muthu M -au721421106014

Karthick Prakash D -au721421106028

Abinash A -au721421106001

Over view:

The Public Transport Optimization using IoT project is a groundbreaking initiative aimed at revolutionizing urban mobility. By integrating IoT sensors and devices into public transportation systems, this project offers a multifaceted solution to long-standing challenges. Real-time tracking of vehicles, predictive analytics for maintenance, and dynamic route adjustments based on traffic and weather data enhance operational efficiency and reduce delays. Passengers benefit from real-time information on schedules, delays, and digital ticketing options, while safety and security are bolstered through surveillance cameras and panic buttons



Objectives:

Efficiency Improvement: Enhance the efficiency of public transportation systems by monitoring and managing vehicle routes, schedules, and passenger loads in real-time.

Reducing Congestion: IoT can help reduce traffic congestion by providing real-time data on traffic conditions and enabling dynamic adjustments to routes and schedules.

Cost Reduction: Minimize operational costs through predictive maintenance of vehicles, fuel consumption optimization, and staff management based on data insights.

Enhancing Safety: Improve passenger and driver safety by monitoring vehicle conditions, detecting accidents or breakdowns, and ensuring timely emergency responses.

IOT SENSORS:

GPS Sensors: Global Positioning System (GPS) sensors are crucial for tracking the location of buses, trains, and other vehicles in real time. They provide accurate geolocation data, which helps in route optimization, schedule adherence, and passenger information systems.

Vehicle Health Sensors: These sensors monitor the condition of public transport vehicles, including engine performance, fuel efficiency, tire pressure, and maintenance needs. They ensure that vehicles are in optimal working condition, reducing breakdowns and delays.

Passenger Counting Sensors: Sensors placed at vehicle entrances and exits use technologies like infrared, ultrasonic, or cameras to count passengers. This data helps in load balancing, scheduling, and optimizing bus or train capacity.

Environmental Sensors: Sensors measuring environmental conditions such as air quality, temperature, humidity, and noise levels can help improve passenger comfort and safety. For instance, they can trigger climate control adjustments or provide alerts in case of poor air quality.

Real time transit information platform:

Real-time Passenger Information: Passengers can access accurate and up-to-date information about transit schedules, delays, and expected arrival times via mobile apps, websites, or digital displays at bus stops and transit stations.

Digital Ticketing: Passengers can purchase tickets, access fare information, and plan their journeys seamlessly through integrated digital ticketing systems, reducing the need for physical ticketing infrastructure.

Enhanced Passenger Safety: Surveillance cameras and panic buttons on vehicles and at transit stops provide added security for passengers, and incidents can be monitored and responded to in real time.

Fleet Management: Transit authorities can track the location and conditions of their vehicles, optimize routes, and improve fuel efficiency, leading to cost savings and reduced environmental impact.

Integration Approach:

IOT Sensor Deployment: Install IoT sensors and devices on public transport vehicles. These sensors should collect data on vehicle location, conditions, passenger load, and other relevant parameters.

Data Aggregation: Set up a central data aggregation and processing system. This system collects data from IoT sensors, passenger apps, and other sources to create a comprehensive dataset.

Connectivity Infrastructure: Establish a robust connectivity infrastructure, including wireless networks, cellular or satellite communication, and cloud-based data storage to ensure real-time data transmission and storage.

Data Processing and Analytics: Implement data processing and analytics tools to transform raw data into actionable insights. Use predictive analytics to anticipate maintenance needs and dynamic routing algorithms to optimize routes.

Passenger Information System: Develop passenger information systems, such as mobile apps and digital signage, to provide real-time information on transit schedules, delays, and vehicle locations.

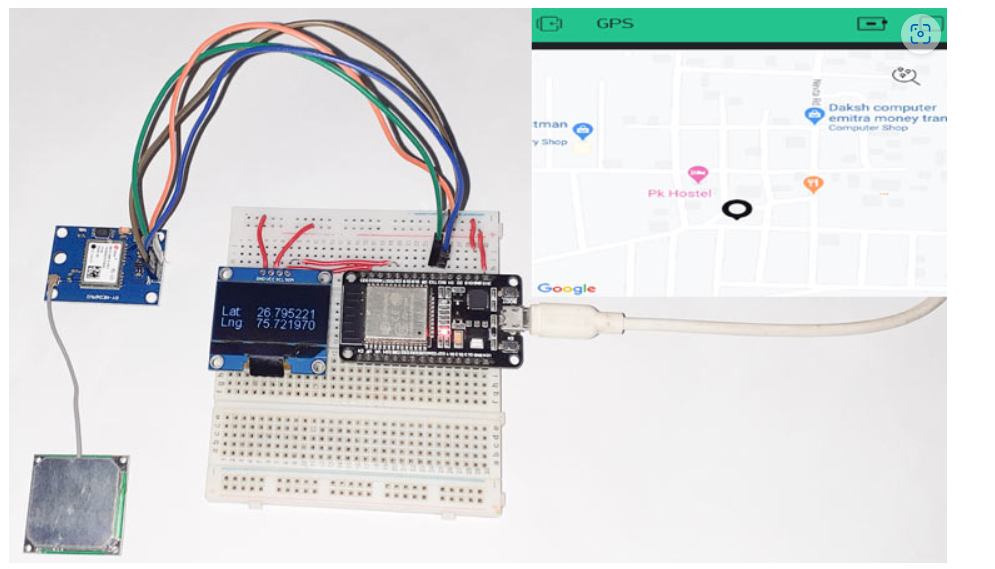
Algorithm for congestion prediction:

Data Collection: install IoT sensors in buses, train stations, and key locations to monitor factors like passenger counts, weather, traffic conditions, and vehicle speed.Collect real-time data, including GPS coordinates, passenger boarding/alighting rates, and any relevant environmental data.

Data Preprocessing: Clean and preprocess the collected data, handling missing values and outliers. Normalize or scale the data as necessary.

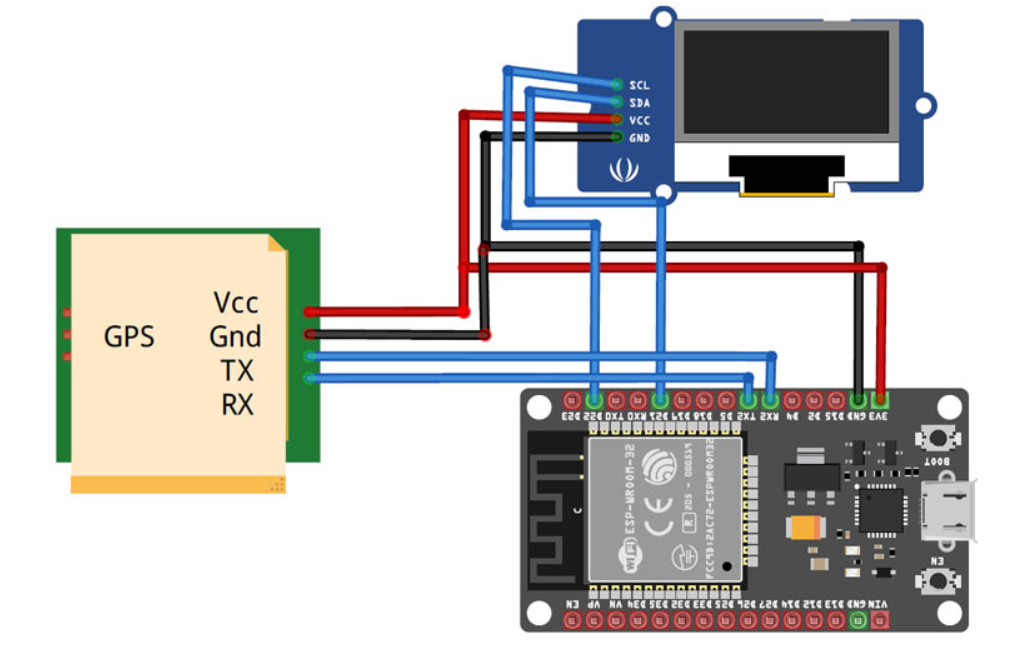
Feature Engineering: Extract relevant features from the data, such as historical congestion patterns, time of day, day of the week, and special events (e.g., concerts, festivals).

Data Fusion: Combine data from different sources (sensors, traffic databases, weather APIs) to create a comprehensive dataset.



Congestion prediction and visualization:

Congestion prediction using IoT and machine learning optimizes public transport by using real-time data to forecast and manage traffic issues. It involves collecting data on passenger counts, vehicle locations, weather, and traffic conditions. This allows transport operators to adjust schedules and routes to reduce congestion, resulting in improved service reliability. Passengers also benefit from real-time alerts, enabling better journey planning. The integration of IoT, data analytics, and machine learning enhances public transportation, providing a smoother commuting experience for all

Circuit diagram: cc

Components required:

•ESP32

•GPS Module

•OLED Display Module

•Jumper Wires

•Breadboard

Specifications for components:

•OLED Driver IC: SSD1306

•Resolution: 128 x 64

•Visual Angle: >160°

•Input Voltage: 3.3V ~ 6V

•Pixel Colour: Blue

Working of ESP32 GPS tracking system:

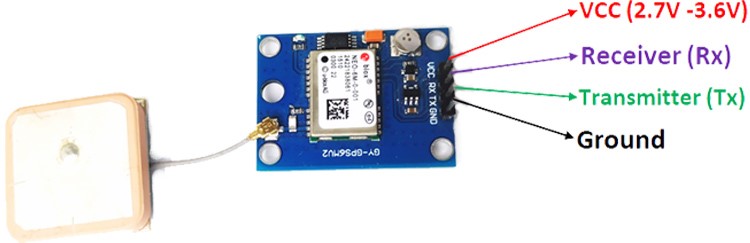
Once the hardware and the program are ready, upload the GPS tracking program into your ESP32 Board. Here Arduino IDE is used to upload the ESP32 GPS NEO 6M code to ESP32 board, so connect the ESP32 to your laptop with a Micro USB Cable and hit the upload button. Once the code is uploaded, the OLED will display the Latitude and Longitude values

A working video and code for this IoT based ESP32 GPS Tracker are given below.

Code:

#include <TinyGPS++.h>  
#include <HardwareSerial.h>  
#include <WiFi.h>  
#include <Wire.h>                 
#include<SH1106.h>   
#include <BlynkSimpleEsp32.h>  
float latitude , longitude;  
String  lat\_str , lng\_str;  
const char \*ssid =  "Galaxy-M20";       
const char \*pass =  "ac312129";   
char auth[] = "loPrSaL0eQFY9clcQ518R1SmYsRVC0eV";   
WidgetMap myMap(V0);   
SH1106 display(0x3c, 21, 22);  
WiFiClient client;  
TinyGPSPlus gps;  
HardwareSerial SerialGPS(1);  
void setup()  
{  
  Serial.begin(115200);  
  Serial.println("Connecting to ");  
  Serial.println(ssid);  
  WiFi.begin(ssid, pass);  
  while (WiFi.status() != WL\_CONNECTED)  
  {  
    delay(500);  
    Serial.print(".");                 
  }  
  Serial.println("");  
  Serial.println("WiFi connected");  
  display.init();  
  display.flipScreenVertically();  
  display.setFont(ArialMT\_Plain\_10);  
  SerialGPS.begin(9600, SERIAL\_8N1, 16, 17);  
  Blynk.begin(auth, ssid, pass);  
  Blynk.virtualWrite(V0, "clr");   
}  
void loop()  
{  
  while (SerialGPS.available() > 0) {  
    if (gps.encode(SerialGPS.read()))  
    {  
      if (gps.location.isValid())  
      {  
        latitude = gps.location.lat();  
        lat\_str = String(latitude , 6);  
        longitude = gps.location.lng();  
        lng\_str = String(longitude , 6);  
        Serial.print("Latitude = ");  
        Serial.println(lat\_str);  
        Serial.print("Longitude = ");  
        Serial.println(lng\_str);  
        display.clear();  
        display.setTextAlignment(TEXT\_ALIGN\_LEFT);  
        display.setFont(ArialMT\_Plain\_16);  
        display.drawString(0, 23, "Lat:");  
        display.drawString(45, 23, lat\_str);  
        display.drawString(0, 38, "Lng:");  
        display.drawString(45, 38, lng\_str);  
        Blynk.virtualWrite(V0, 1, latitude, longitude, "Location");  
        display.display();  
      }  
     delay(1000);  
     Serial.println();    
    }  
  }    
  Blynk.run();  
}

Neo 6M GPS module:

The NEO-6M module comes with a dimension of 16 x 12.2 x 2.4 mm package. It has 6 Ublox positioning engines offering unmatched performance. It is a good performance GPS receiver with a compact architecture, low power consumption, and reliable memory options. It is ideal for battery-operated mobile devices considering its architecture and power demands. The Time-to-First-Fix is less than 1 second and it enables it to find the satellites almost instantly. The output is in the format of NMEA standards, which can be decoded to find the coordinates and Time of the location. 

Components required:

Power supply: 2.8V to 5V

Interface: RS232 TTL

Built-in EEPROM and external antenna

Default baud rate: 9600 bps

Platform:

* Cisco Kinetic for Cities: Cisco offers a platform that focuses on smart city solutions, including public transport optimization. It allows for real-time data collection and analytics to enhance transportation operations and improve the overall urban experience.
* Siemens Mobility: Siemens provides solutions for intelligent traffic management and public transport optimization, including IoT-based systems for efficient scheduling, predictive maintenance, and real-time passenger information.
* Moovit: Now part of Intel, offers an IoT-driven platform for public transit agencies to optimize routes, schedules, and real-time passenger information, as well as deliver data-driven insights.
* HERE Technologies: HERE offers a location-based platform that can be applied to public transport optimization, including route planning, real-time tracking, and predictive maintenance.
* TransLoc: TransLoc, now a part of Ford Smart Mobility, specializes in transit technology, offering solutions for route planning, on-demand services, and passenger information systems.
* Optibus: Optibus provides a cloud-based platform that uses IoT data for public transport optimization, including scheduling, routing, and demand-responsive services

Web development platform:

Front-End Development: Use popular web development frameworks like React or Angular for user-friendly interfaces displaying real-time transport data.

Back-End Infrastructure: Employ Node.js or Python to build a robust back-end that processes and manages IoT data from sensors and devices.

Database: Utilize databases such as MongoDB or PostgreSQL to store and retrieve IoT-generated data for analytics and decision-making.

API Integration: Create APIs to connect with IoT sensors, vehicle tracking systems, and external data sources for real-time updates.

Security and Scalability: Prioritize security measures and ensure scalability to handle large volumes of data, offering a responsive and secure web platform for public transport optimization

Code for website:

Front-end developer(HTML,CSS,Javascript)

<!DOCTYPE html>

<html>

<head>

<title>Public Transport Tracker</title>

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

</head><body>

<div id="map">

</div>

<script src="script.js">

</script></body>

</html>

// Initialize the Map

mapboxgl.accessToken = 'YOUR\_MAPBOX\_ACCESS\_TOKEN’;

const map = new mapboxgl.Map

({

container: 'map’,

style: 'mapbox://styles/mapbox/streets-v11’,

center: [-74.006, 40.7128], // Example: New York City coordinates

zoom: 12,

});

// Function to add a marker for a vehiclefunction addMarker(coordinates)

{ new mapboxgl.Marker().setLngLat(coordinates).addTo(map);

}

// Simulate real-time updates (replace with actual data source)

setInterval(() => {

const latitude = 40.7128 + (Math.random() - 0.5) \* 0.01; // Simulated latitude change

const longitude = -74.006 + (Math.random() - 0.5) \* 0.01; // Simulated longitude change

const coordinates = [longitude, latitude];

// Remove previous markers to show only the latest position map.getCanvas().querySelectorAll(‘.m

apboxgl-marker').forEach((marker) =>

marker.remove()); addMarker(coordinates);

}, 5000); // Update every 5 seconds (adjust as

 needed)

Back-end(node.js)

npm install express

const express = require('express’);

const app = express();

app.use(express.static('public')); // Serve your

front-end files from the "public" directory

const PORT = process.env.PORT || 3000;

app.listen(PORT, () => {

console.log(`Server is running on port ${PORT}`);

});

Results and analysis:

Public transport optimization through IoT project analysis has emerged as a groundbreaking solution to enhance the efficiency and convenience of urban transportation systems.

By leveraging real-time data from interconnected sensors and devices on buses, trams, and subway systems, this innovative approach allows transportation authorities to monitor and analyse critical information such as vehicle location, passenger load, traffic conditions, and maintenance requirements.

This wealth of data enables decision-makers to make informed adjustments to routes and schedules, minimizing congestion, reducing wait times, and ultimately providing a more reliable and sustainable mode of public transit. As a result, commuters experience improved accessibility, reduced travel times, and decreased environmental impact, making cities more liveable and connected.

IoT-based public transport optimization has the potential to revolutionize urban mobility, paving the way for smarter, more efficient, and eco-friendly transportation systems

THANK YOU