Internet Of Things

Public transport optimization using IOT sensors

Phase 2: Innovation

Introduction:

Smart public transportation has become one of the most important things for developing cities and improving people's quality of life. Public transportation users face many problems, the most important of which is the long wait at the bus station. The main objective of the prototype is to reduce the wait time at the bus station by knowing the nearest buses to a user, the real-time location of buses on the Google map to help passengers track buses in real-time, the arrival time of buses, and speed. The system was implemented based on Internet of Things (IoT) technology, by using the Global Positioning System (GPS), a microcontroller with a built-in Wi-Fi module (ESP32).

Steps involved in this project in the following:

1.Connect ESP32 with gps sensor and fit in public vehicle

2.Code ESP32 in python to obtain data from gps sensor

3.Send data to the server to store it on database.

4.Display location of vehicle in the website by get the data stored in database.

Let us see each step in detail

1.Connect ESP32 with gps sensor and fit in public vehicle

Firstly, you'll need to connect your GPS module to the ESP32. You can use the library in python to interact with the GPS module. i.e.,Obtain the data of gps sensor and calculate the latitude and longitude of the moving vehicle then

set up your ESP32 with a GPS module to send latitude and longitude data to a web server via the http request .

2.Code ESP32 in python to obtain data from gps sensor

Firstly code ESP32 to retreive data from gps sensor then you'll need to install the urequests library on your ESP32 to make HTTP requests.so that you send post requests to web server . You can do this using the ampy tool:

ampy -p /dev/tty.SLAB_USBtoUART put urequests.py

Then, you can use the following MicroPython code:

PROGRAM:

```
import machine
import ubinascii
import urequests
uart = machine.UART(2, baudrate=9600, rx=16, tx=17)
def parseGPS(data):
sdata = data.split(",")
if sdata[0] == "$GPGGA":
if sdata[2] != '':
lat = decode(sdata[2])
lon = decode(sdata[4])
return (lat,lon)
def decode(coord):
l = list(coord)
```

```
 for i in range(0,len(l)-1):
if l[i] == ".":
break
base = l[0:i-2]
degi = l[i-2:len(l)]
baseint = int("".join(base))
degint = int("".join(degi))
degfrac = degint / pow(10, len(degi))
return baseint + degfrac/60


while True:
if uart.any():
gps_data = uart.readline()
coord = parseGPS(gps_data)
if coord:
url = 'http://your-web-server.com/api/gps'
headers = {'content-type': 'application/json'}
data = '{"latitude": "%s", "longitude": "%s"}' % coord
        resp = urequests.post(url, data=data, headers=headers)
```

This script reads GPS data from the uart ,parses the data from the GPGGA sentences find latitude and longitude and send this data to the web server.

3.Send data to the server to store it on database

Create a Flask web server that receives GPS data and stores it in a SQLite database. This example assumes that the GPS data is sent as a JSON object with ‘latitude’ and ‘longitude’ fields.

First, install the necessary Python packages:

```
pip install flask flask_sqlalchemy
```

Then, you can use the following Python code:

PROGRAM:

```
from flask import Flask, request
from flask_sqlalchemy import SQLAlchemy

app = Flask(__name__)
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:////tmp/test.db'
db = SQLAlchemy(app)

class GPSData(db.Model):
id = db.Column(db.Integer, primary_key=True)
latitude = db.Column(db.Float)
longitude = db.Column(db.Float)

@app.route('/api/gps', methods=['POST'])
def post_gps_data():
data = request.get_json()
    gps_data = GPSData(latitude=data['latitude'], longitude=data['longitude'])
db.session.add(gps_data)
```

```
 db.session.commit()
return '', 204


if __name__ == '__main__':
db.create_all()
    app.run(debug=True)
```

This script creates a new SQLite database in '/tmp/test.db' with a single table for GPS data. It defines one route '/api/gps' that accepts POST requests. The posted JSON data is extracted from the request, stored in the database, and a 204 No Content response is sent back.

4.Display location of vehicle in the website by get the data stored in database.

To display the live location of the vehicle on a map on your website, we use JavaScript along with a mapping library like Leaflet.js. Here's a basic example of how you might do this:

First, include the necessary JavaScript libraries in your HTML:

PROGRAM:

```
<!DOCTYPE html>
<html>
<head>
<title>Live Vehicle Tracking</title>
   <link rel="stylesheet" href="https://unpkg.com/leaflet/dist/leaflet.css" />
<script src="https://unpkg.com/leaflet/dist/leaflet.js"></script>
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
```

```
</head>
<body>
<div id="mapid" style="height: 600px;"></div>

<script>
var mymap = L.map('mapid').setView([0, 0], 2);
   L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
maxZoom: 19,
}).addTo(mymap);

var marker = L.marker([0, 0]).addTo(mymap);

function updateMarker() {
$.getJSON('/api/gps', function(data) {
var lat = data['latitude'];
var lon = data['longitude'];
marker.setLatLng([lat, lon]);
mymap.setView([lat, lon], 13);
});
}

setInterval(updateMarker, 5000);
</script>
</body>
</html>
```

This script sets up a map and places a marker on it. Every 5 seconds, it makes a GET request to '/api/gps' to get the latest GPS data and update the marker's position accordingly.You'll need to implement the '/api/gps' route in your Flask application to return the latest GPS data from your database

PROGRAM:

```
@app.route('/api/gps', methods=['GET'])
def get_latest_gps_data():
latest_data = GPSData.query.order_by(GPSData.id.desc()).first()
    return {'latitude': latest_data.latitude, 'longitude': latest_data.longitude}
```

Remember to replace'https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png' with the tile URL of your preferred map provider if you're not using OpenStreetMap. You might also want to adjust the initial view and zoom level of the map depending on your needs.this document gives the general idea about the project,through the development of this project a clear algorithm is made to collect information from esp32 and track location accurately in the frontend of the website with user friendly user interface. So now a user can see the live location of public transport vehicle in website it will be improve public transport optimization