PUBLIC TRANSPORT OPTIMIZATION :

Phase-3: Development part 1

INTRODUCTION:

In recent years, the Internet of Things (IoT) has emerged as a transformative technology with the potential to revolutionize various aspects of our lives, including public traffic optimization. IoT in public traffic optimization refers to the integration of smart, interconnected devices, sensors, and data analytics to improve the efficiency, safety, and sustainability of transportation systems within urban areas. This concept has gained significant attention as cities around the world grapple with growing traffic congestion, environmental concerns, and the need for more sustainable and accessible transportation options.

⍰COMPONENTS REQUIRED:

⍰ Arduino Board

⍰GPS Module (e.g., NEO-6M or NEO-7M)

GSM Module (e.g., SIM800L or SIM900)

HARDWARE DEVELOPMENT :

Developing hardware for an Arduino-based system for public transport optimization can be an exciting and impactful project. Here's an overview of the key hardware components and considerations for such a project:

1. *Arduino Board*: Choose an appropriate Arduino board based on your project's requirements. Popular choices include the Arduino Uno, Arduino Mega, or Arduino Nano. The selection depends on factors like the number of sensors and peripherals you plan to connect and the power requirements.

2. *Sensors*:

- *GPS Module*: Integrating a GPS module enables real-time tracking and monitoring of vehicle locations. This data is crucial for optimizing routes and schedules.

- *Proximity Sensors*: Ultrasonic or infrared sensors can be used to detect the presence of passengers at bus stops or within vehicles.

- *Environmental Sensors*: To monitor temperature, humidity, and air quality inside vehicles.

- *Camera Modules*: For capturing images and video data, which can be used for security, traffic analysis, or passenger counting.

- *RFID/NFC Readers*: These can be used for contactless ticketing and passenger identification.

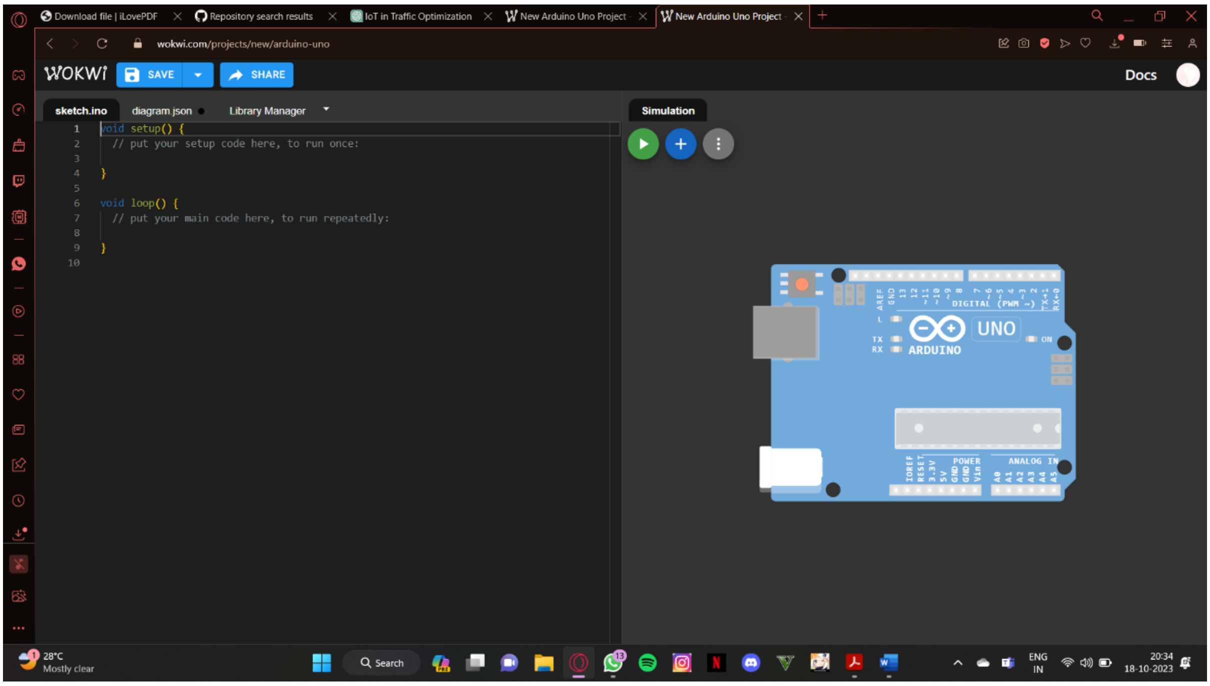
3. *Communication Modules*:

- *GSM/GPRS/3G/4G/5G Module*: For transmitting data to a central server or cloud platform, enabling remote monitoring and control.

- *Wi-Fi Module*: To connect to local Wi-Fi networks for data exchange or remote management.

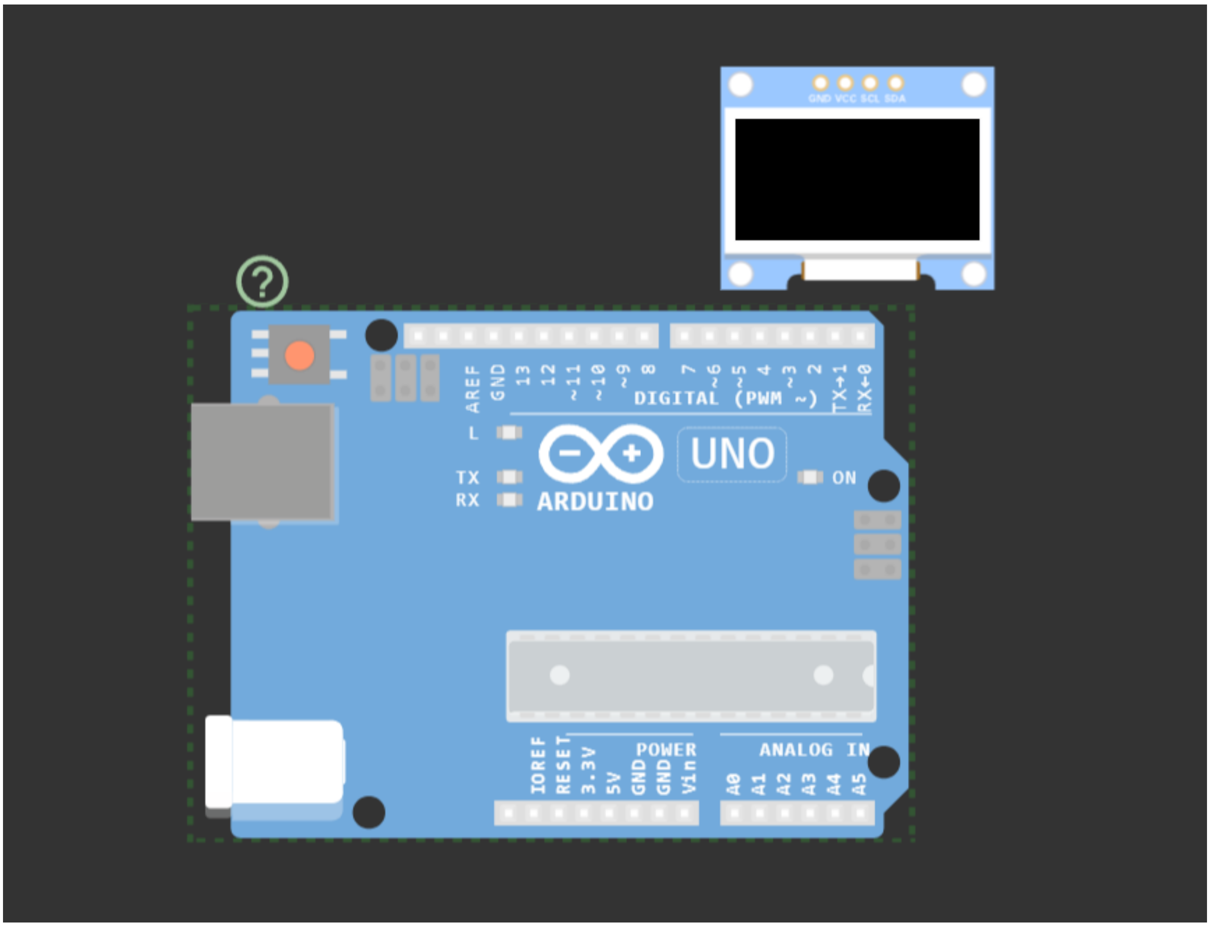
- *Bluetooth Module*: For communication with smartphones or other nearby devices.

STEP-1:

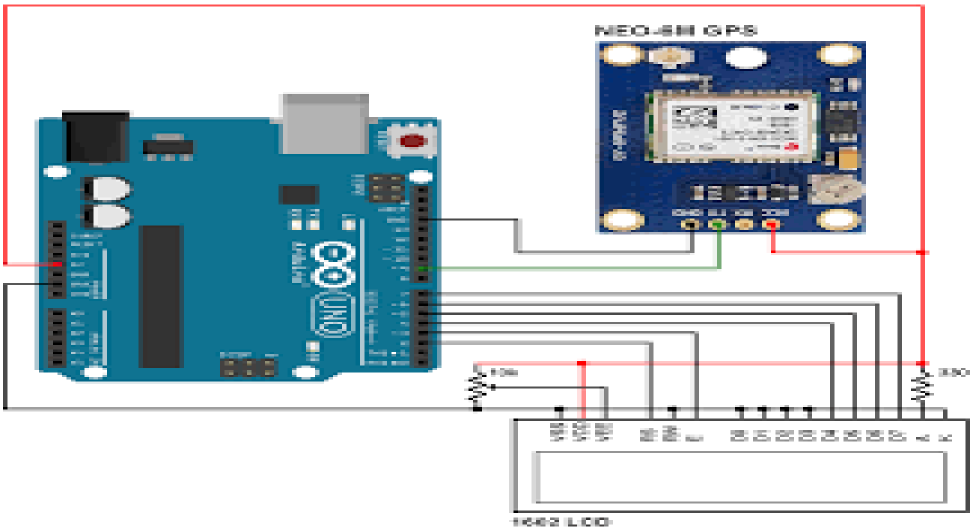


```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

STEP 2:


GND VCC SCL SDA
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM ~)
L
TX
RX
ARDUINO
UNO
ON
IOREF
RESET
3.3V
5V
GND
GND
Vin
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN

Step-3


NEO-6M GPS
10k
330
1602 LCD

SOFTWARE DEVELOPMENT:

```
#include <SoftwareSerial.h>
#include <TinyGPS++.h>
// Define the pins for your GPS and GSM modules
SoftwareSerial gpsSerial(2, 3); // GPS RX, GPS TX
SoftwareSerial gsmSerial(4, 5); // GSM RX, GSM TX
TinyGPSPlus gps;
void setup() {
Serial.begin(9600);
gpsSerial.begin(9600);
gsmSerial.begin(9600);
Serial.println("GPS-GSM Vehicle Tracking");
Serial.println("Initializing GSM module...");
// Initialize GSM module (send AT commands)
gsmSerial.println("AT");
delay(1000);
```

```
if (gsmSerial.find("OK")) {
Serial.println("GSM module is ready.");
Serial.println("Turning on GPS module...");
gsmSerial.println("AT+CGNSPWR=1"); // Turn on GPS
delay(1000);
if (gsmSerial.find("OK")) {
Serial.println("GPS module is ready.");
} else {
Serial.println("GPS module not responding.");
}
} else {
Serial.println("GSM module not responding.");
}
}
void loop() {
while (gpsSerial.available() > 0) {
if (gps.encode(gpsSerial.read())) {
if (gps.location.isValid()) {
// Get GPS data
float latitude = gps.location.lat();
float longitude = gps.location.lng();
// Send GPS data via GSM
gsmSerial.print("AT+CMGS=\"YOUR_PHONE_NUMBER\"\r");
delay(1000);
gsmSerial.print("Latitude: ");
gsmSerial.print(latitude, 6);
gsmSerial.print(", Longitude: ");
gsmSerial.print(longitude, 6);
gsmSerial.write(0x1A);
delay(1000);
Serial.print("Latitude: ");
Serial.print(latitude, 6);
Serial.print(", Longitude: ");
Serial.println(longitude, 6);
}
}
}
}
```

Conclusion:

In conclusion, public transport optimization using the Internet of Things (IoT) represents a significant step forward in the quest for efficient, sustainable, and passenger-friendly urban transportation systems. This technology has the potential to transform the way we plan, manage, and experience public transit. Here are some key takeaways and conclusions regarding public transport optimization with IoT:

Efficiency and Traffic Management: IoT-based systems allow real-time monitoring of traffic conditions, enabling dynamic traffic management to reduce congestion and improve the flow of public transport.

Enhanced Passenger Experience: IoT solutions provide passengers with real-time information on vehicle locations, arrivals, and departures, improving the overall transit experience and passenger satisfaction.

Safety and Security: IoT systems can enhance the safety and security of public transport by enabling real-time surveillance, emergency response, and predictive maintenance, thereby reducing accidents and ensuring passenger security.