DEVELOPMENT OF
REAL-TIME TRANSIT INFORMATION PLATFORM

Phase-4: Continue building the project by developing the real-time transit information platform. Use web development technologies to create a platform that displays real-time transit information. Design the platform to receive and display real-time location, ridership, and arrival time data from IoT sensors

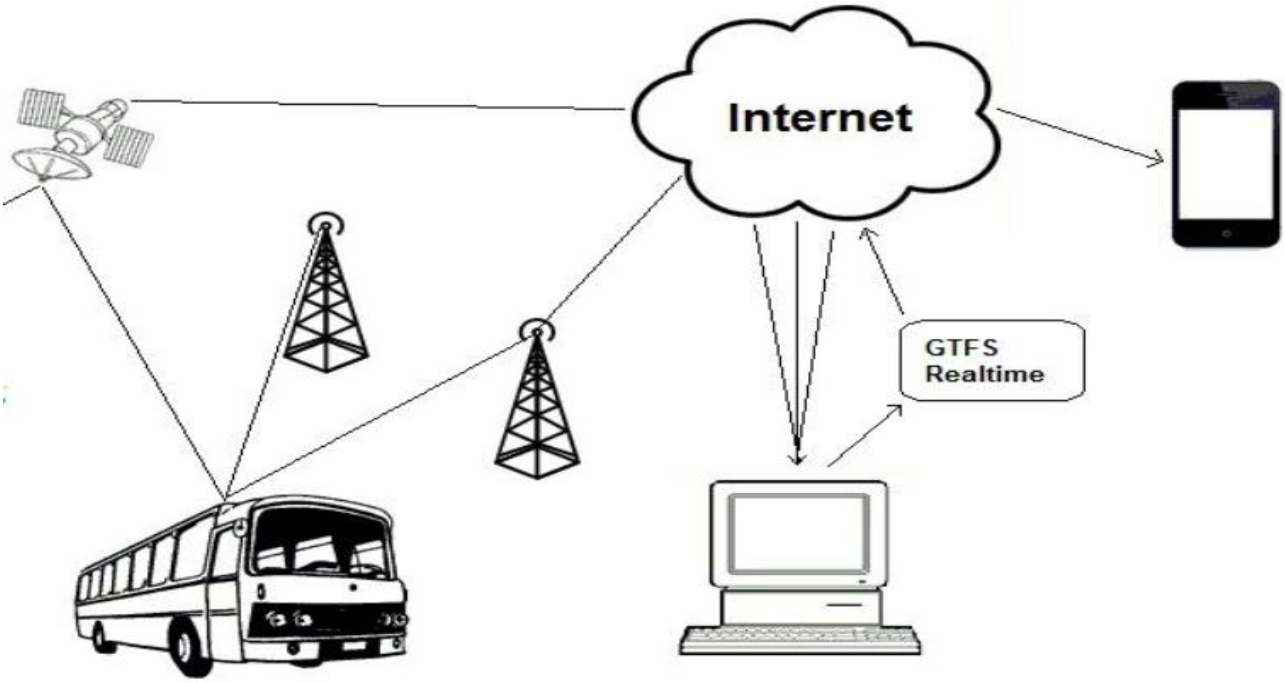
INTRODUCTION:

As cities evolve and urban populations surge, efficient and reliable public transportation systems have become paramount for sustainable urban living. In this dynamic landscape, a real-time transit information platform emerges as a transformative solution, seamlessly connecting commuters to the pulse of their city's transit network. Leveraging the power of web development technologies, such as HTML, CSS, and JavaScript, this platform becomes the digital gateway to the world of public transit, providing instant access to real-time location, ridership, and arrival time data.

The driving force behind this endeavor is the Internet of Things (IoT), where a web of sensors on buses, trams, and subways gathers and transmits live data to our platform. This intricate dance of data exchange allows commuters and transit authorities alike to gain a deeper understanding of their urban transit ecosystems, ultimately leading to more informed decisions and improved travel experiences.

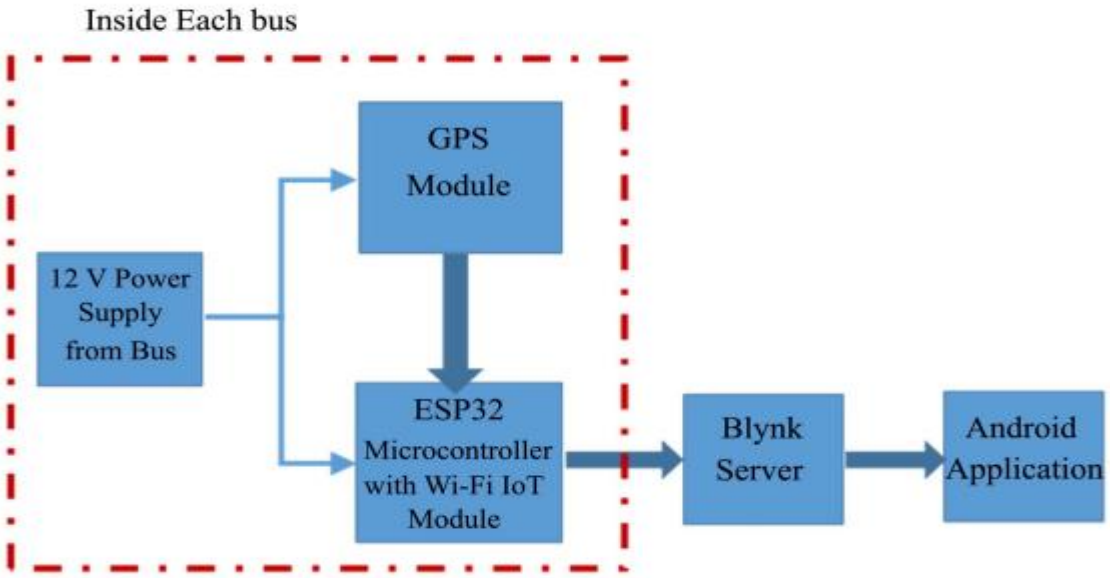
So our aim is to create this real-time transit information platform, from its inception as an idea to its realization as a functional web application. We'll dive into the fundamental technologies, design principles, and implementation strategies that will empower you to

build a system capable of receiving and displaying real-time transit data from IoT sensors.



SYSTEM DESIGN:

The system was implemented based on Internet of Things (IoT) technology, by using the Global Positioning System (GPS), a micro-controller with a built-in Wi-Fi module (ESP32), and a mobile user interface by the Blynk-IoT platform. All data obtained is displayed by the GPS sensors for bus locations (longitude and latitude) and speed on the smartphone application. The distance between the bus location and the passenger that will appear in the mobile app.



The consists of an Android application designed for users who want real-time information about the buses. The app will display information about buses such as real-time location on Google Maps, speed, distance, and arrival time of each bus. The proposed system includes an ESP32 with a Wi-Fi built-in module, a GPS module, and an Android app connected to the server. The proposed system is operated by GPS and ESP32, which are installed in each bus with a power supply that may be obtained from the bus. GPS receives the satellite signals and then the position coordinates with latitude, longitude and speed are determined for moving buses. After receiving the data, the tracking data can be transmitted using wireless communications systems. In this system, the ESP32 is a micro-controller with a Wi-Fi module. All the information collected by ESP32, such as location (latitude and longitude), speed, etc., will be uploaded to the Blynk Server. Based on IoT, the user can access this information on a bus through the Android application.

STEPS INVOLVED:

1. **SETTING UP BLYNK SERVER:**Install and configure a Blynk server on a host or a cloud-based server. Obtain the necessary credentials, including the server address, port, and authentication token for your Blynk server. These credentials will be used by your devices to connect to the server.

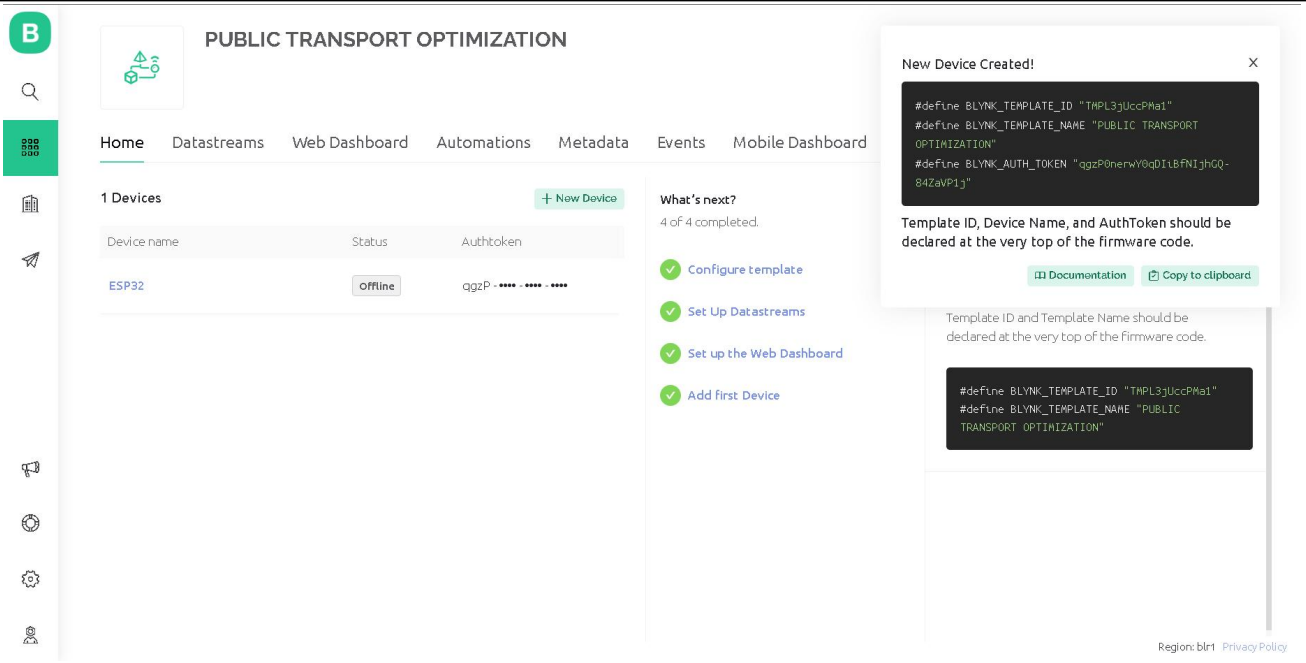
CONFIGURATION RASPBERRY PI AND ESP32:

- Install the Blynk library on your Raspberry Pi and ESP32. We can find the Blynk library and documentation for your respective platforms on the Blynk website.
- Write code on your Raspberry Pi and ESP32 to collect GPS and ridership monitoring data and transmit it to the Blynk server using the Blynk library. Use the provided authentication token to connect to your Blynk server.
- Set up the appropriate virtual pins in your Blynk project to receive and display the data sent by **your devices. You can configure widgets in your Blynk project to visualize the data.**

CONNECTION ESTABLOSHMENT BETWEEN SENSOR SYSTEM AND BLYNK SERVER:

By using the following code the sensor sytem can be connected to the wifi and can be linked with the Blynk server for data feeding.

```
#include <WiFi.h>
#include <BlynkSimpleEsp32.h>

char auth[] = "qgzP0nerwY0qDIiBfNIjhGQ-84ZaVP1j";
char ssid[] = "My wifi";
char pass[] = "12345678";

BlynkTimer timer;

void setup() {
  Blynk.begin(auth, ssid, pass);
}

void loop() {
  Blynk.run();
}
```



DATA FEEDING FROM GPS TO BLYNK SERVER:

First install blynk python library in ESP32 board using following code,

" pip install blynk-library-python "

Then load the following code to the ESP32 board,

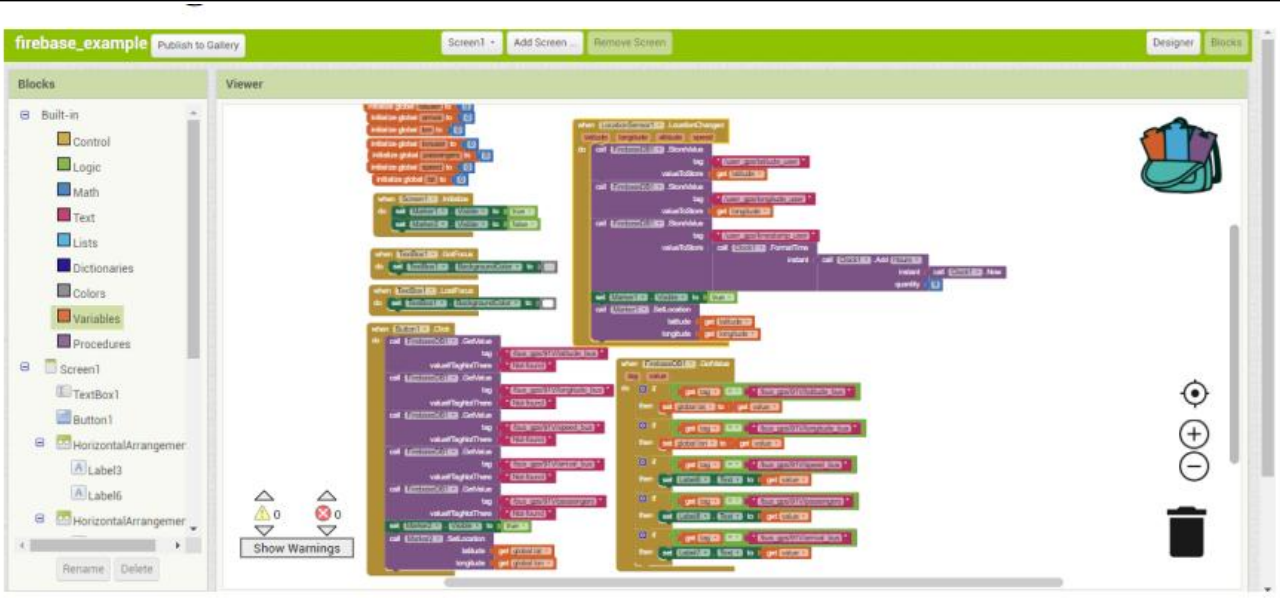
```
import time
from blynkapi.blynk import Blynk

# Your Blynk authentication token
auth_token = 'your_auth_token'

# Initialize the Blynk client
blynk = Blynk(auth_token)

# Function to send GPS data to the Blynk server
def send_gps_data():
```

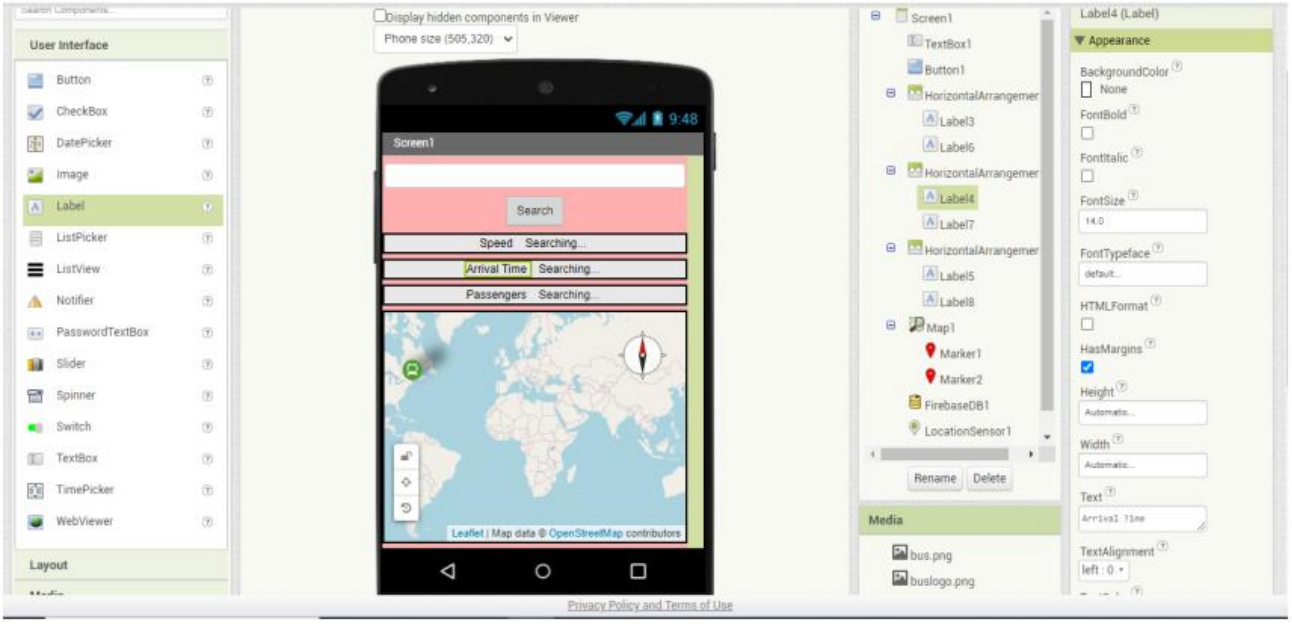
```
        # Replace these values with actual GPS data
        latitude = 123.456
        longitude = 78.910

        # Send latitude and longitude to virtual pins V1 and V2
        blynk.virtual_write(1, latitude)
        blynk.virtual_write(2, longitude)

    # Main loop
    while True:
        try:
            send_gps_data()
        except Exception as e:
            print(f"Error: {str(e)}")

        # Update GPS data every 10 seconds (adjust as needed)
        time.sleep(10)
```

PLATFORM DESIGN USING MIT APP INVENTOR:

- Design the app interface: Use the visual designer to create the user interface for your public platform. You can design screens for viewing GPS data and ridership information.

- Add Blynk components: In MIT App Inventor, add Blynk components that allow your app to communicate with the Blynk server. Configure these components with the server address, port, and authentication token.

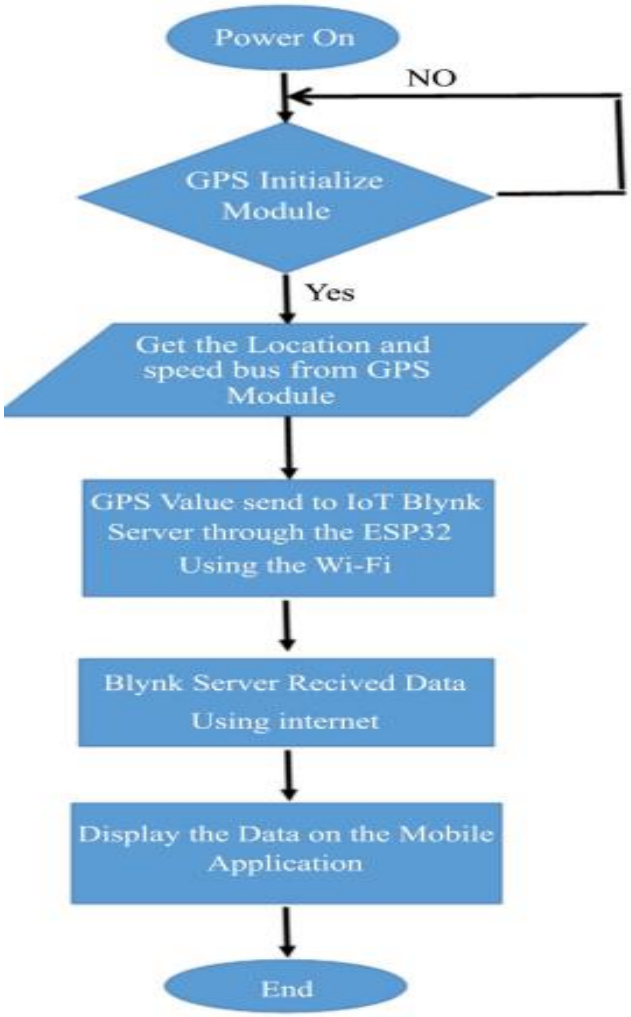




SYSTEM FLOW:

The user must detect his location by activating the location feature in a smartphone. To get the information entered the application will provide the details about buses, bus location, bus speed, bus arrival time, nearest bus from a user by offering the distance between user location and bus. This information will assist the passenger to select their suitable bus.A GPS module connected to an ESP32 Micro-controller with a built-in Wi-Fi module is placed inside each bus. When the power supply is on, the GPS module

communicates continuously with the satellite to get coordinates. The GPS module will initialize itself, then the module will get the coordinates, but if the coordinates are not received, then the module will initialize again. Once the GPS obtains the coordinates, it sends the data, including latitude and longitude, and speed to the IoT Blynk server through the ESP32. At the Blynk server, the latitude and longitude are extracted and used on the visual map in the Blynk application. The live location of the bus can be seen on the Google map. Continuous data digital updates such as speed, distance, and the arrival time of the bus are displayed on the mobile application.



APPLICATION INTERFACE: