**Phase 3**

**PROBLEM, IDENTIFICATION AND DESIGN THINKING**

|  |  |
| --- | --- |
| **Date** | 31-10-2023 |
| **Team Id** | **Proj\_223982\_team** |
| **Project Name** | Public Transport Optimization |
| **Students with**  **Id** | Saiprasanna P  Rajasuriya L K  Naveen Kumar K  Udhayakumar P  Vignesh V |

* **Project overview**

The Public Transport Optimization project aims to enhance the efficiency, accessibility, and sustainability of public transportation systems within a given urban area or region. By employing advanced technology, data analytics, and strategic planning, this project will strive to improve the overall quality of public transport services, reduce congestion, and minimize environmental impact.

* **Purpose and Scope**

1. The Public Transport Optimization project aims to improve the existing public transportation system by creating a more efficient, accessible, and sustainable urban transit network.
2. It aims to reduce traffic congestion, minimize environmental impact, and enhance the transportation experience for residents and visitors.

* **Objectives**

1. Increase the reliability of public transportation services to encourage regular usage.
2. Promote sustainable practices in the operation and maintenance of public transportation services.
3. Financial Sustainability: Develop a sustainable financial model for the optimized public transport system.

* **Hardware Components**
* ESP32 Microcontroller
* Ultrasonic Sensors
* **Component Descriptions**

1. ESP32 Microcontroller: The ESP32 is a versatile microcontroller that provides Wi-Fi connectivity, making it suitable for IoT applications. In this project, it acts as the central processing unit.
2. Ultrasonic Sensor: The ultrasonic sensor is responsible for measuring passenger’s count.

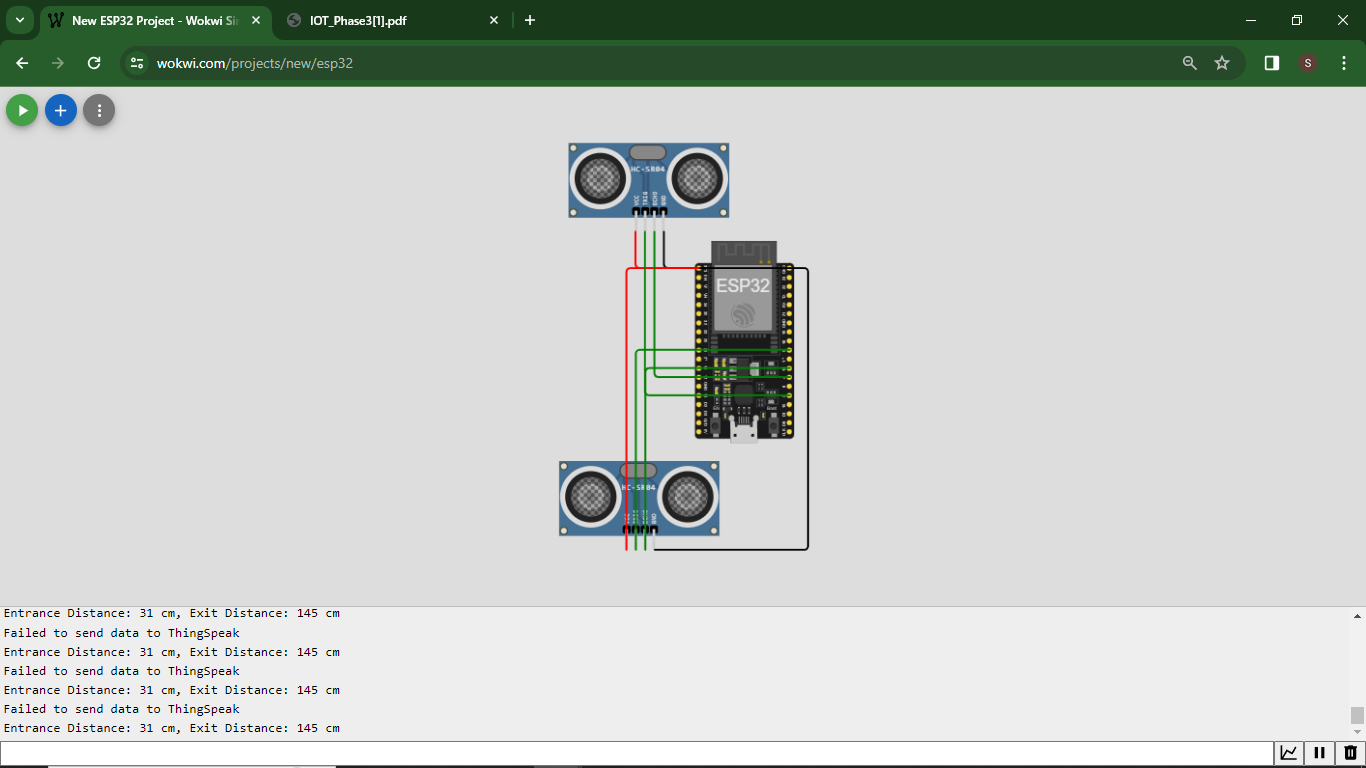
* **Software Components**

1. DHTesp library for DHT22 sensor.
2. Ultrasonic library for ultrasonic sensor.
3. HingSpeak library for cloud integration.

* **Software Descriptions**

1. **DHTesp Library:** This library enables communication with the DHT22 sensor, making it possible to read temperature and humidity values.
2. **Ultrasonic Library:** The Ultrasonic library provides functions for interfacing with the ultrasonic sensor, allowing accurate Passenger’s count measurements.
3. **ThingSpeak Library:** The ThingSpeak library is used for sending data to the ThingSpeak platform, where it can be visualized and monitored.

* **System Architecture**
* The system architecture involves the ESP32 as the central controller, which interfaces with the DHT22 and Ultrasonic sensors. The ESP32 collects data from these sensors, processes it, and sends the data to the ThingSpeak platform over Wi-Fi. ThingSpeak then displays the data.



* **Code**

#include <Ultrasonic.h>

#include <WiFi.h>

#include <ThingSpeak.h>

Ultrasonic entranceSensor(2, 4);  // Trigger (D2) and Echo (D4) pins for the entrance sensor

Ultrasonic exitSensor(5, 16);     // Trigger (D5) and Echo (D16) pins for the exit sensor

int incomingPassengers = 0;

int outgoingPassengers = 0;

bool entranceDetected = false;

bool exitDetected = false;

const char\* ssid = "Wokwi-GUEST"; // Replace with your Wi-Fi SSID

const char\* password = ""; // Replace with your Wi-Fi password

const char\* server = "api.thingspeak.com";

const unsigned long channelID = 2325427; // Your ThingSpeak Channel ID

const char\* writeAPIKey = "FIAPRTHYVSVZKM2X"; // Your ThingSpeak Write API Key

WiFiClient client;

void setup() {

**Serial**.begin(115200);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL\_CONNECTED) {

    delay(1000);

**Serial**.println("Connecting to WiFi...");

  }

**Serial**.println("Connected to WiFi");

}

void loop() {

  long entranceDistance = entranceSensor.read();

  long exitDistance = exitSensor.read();

**Serial**.print("Entrance Distance: ");

**Serial**.print(entranceDistance);

**Serial**.print(" cm, Exit Distance: ");

**Serial**.print(exitDistance);

**Serial**.println(" cm");

  if (entranceDistance < 30 && !entranceDetected) {

    incomingPassengers++;

    entranceDetected = true;

**Serial**.println("Passenger entered: " + String(incomingPassengers));

  } else if (entranceDistance >= 30) {

    entranceDetected = false;

  }

  if (exitDistance < 30 && !exitDetected) {

    outgoingPassengers++;

    exitDetected = true;

**Serial**.println("Passenger exited: " + String(outgoingPassengers));

  } else if (exitDistance >= 30) {

    exitDetected = false;

  }

  // Send passenger counts to ThingSpeak

  ThingSpeak.begin(client);

  ThingSpeak.setField(1, incomingPassengers);

  ThingSpeak.setField(2, outgoingPassengers);

  int status = ThingSpeak.writeFields(channelID, writeAPIKey);

  if (status == 200) {

**Serial**.println("Data sent to ThingSpeak successfully");

  } else {

**Serial**.println("Failed to send data to ThingSpeak");

  }

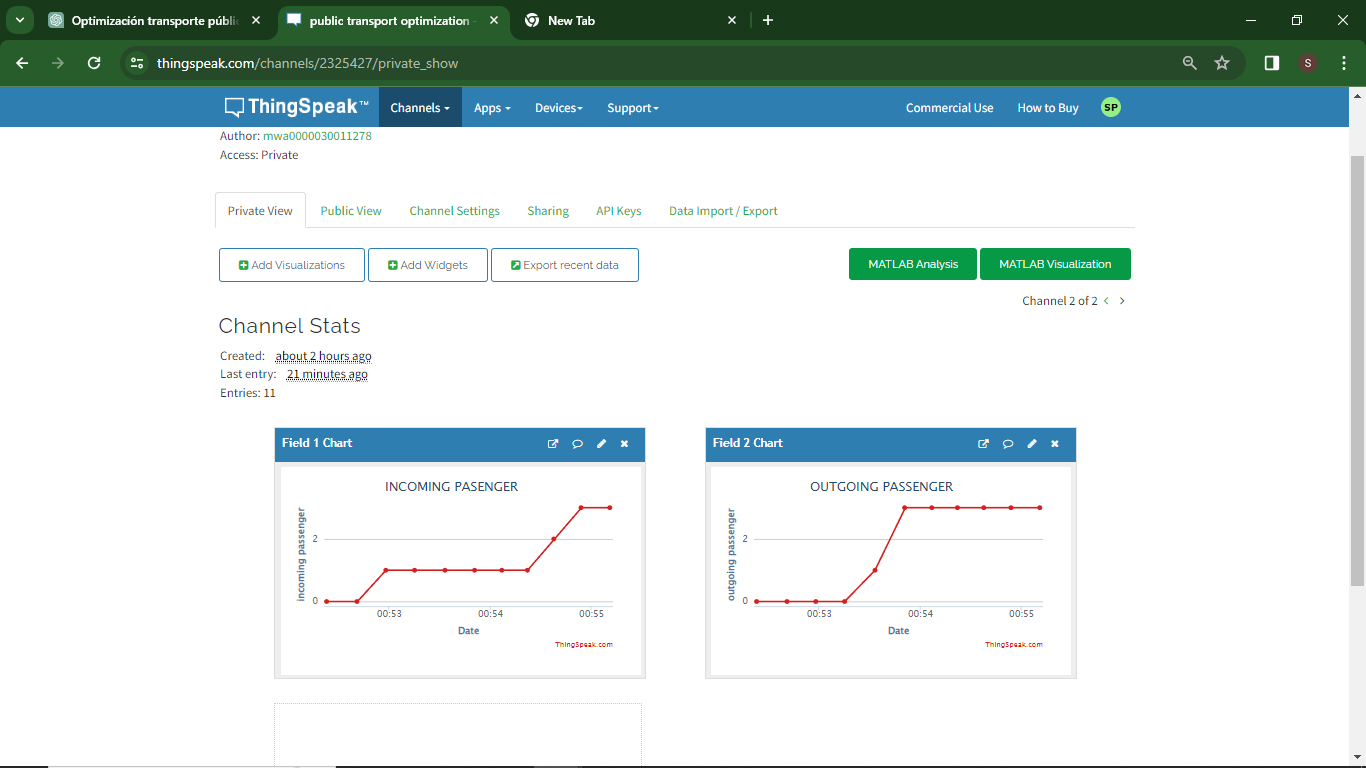
  delay(1000);  // Adjust the delay as needed

}

* **Operation**

1. The system operates by continuously reading passenger’s count and total capacity data from the DHT22 and Ultrasonic Sensors. It sends this data to ThingSpeak via WiFi for remote monitoring.
2. The significance of the Public Transport Optimization is defining the limit at which warning during extra passengers.

* **Outputs**



* **Conclusion**

It proposed Public Transport Improvement Program addresses challenges in management, enhancing passenger experience through real-time seat availability and control technology. It prioritizes safety, streamlines logistics, and promotes flexibility. The program also emphasizes sustainability and efficiency, reducing carbon emissions and creating an environmentally friendly system.