

INTERNSHIP PROJECT REPORT

**On**

#### IOT-BASED SMART LUGGAGE SYSTEM

Submitted by:

E.Raghavi

P.Vaishnavi

J.Rebakha Sharon

G.Bhavana

Under the guidance of Prof. T. Kishore Kumar

Professor, Department of ECE

NIT Warangal

**Centre for Training and Learning** NATIONAL INSTITUTE OF TECHNOLOGY WARANGAL **HANAMKONDA - 506004, TELANGANA, INDIA**

## BONAFIED CERTIFICATE

This is to certify that this project report entitled **“IOT-BASED SMART LUGGAGE SYSTEM”** submitted to National Institute of Technology, Warangal and National Institute of Technology, Warangal, is a bonafide record of work done by **“E.Raghavi , P.Vaishnavi , J.Rebakha Sharon , G.Bhavana”** under my supervision from **20 May 2024** to **20 Jun 2024**

#### Supervisor

Prof. T. Kishore Kumar Professor, Department of ECE

NIT Warangal

Place: Warangal Date: 20 JUNE 2024

## DECLARATION

This is to declare that this report has been written by us. No part of the report is plagiarized from other sources. All information included from other sources have been duly acknowledged. We aver that if any part of the report is found to be plagiarized, we are shall take full responsibility for it.

Place: Warangal Date: 20 Jun 2024

## 

## ACKNOWLEDGEMENT

We extend our sincere gratitude to the following individuals and organizations whose support and contributions were instrumental in the successful completion of this project and thesis.

First and foremost, we express our deepest appreciation to Dr. T. Kishore Kumar, Professor and Head of CTL (Center for Training and Learning), NIT Warangal. His constant motivation, guidance, and unwavering support were pivotal in helping us achieve this significant milestone.

The NIT Warangal AIOT (Artificial Intelligence of Things) and Its Applications Internship, conducted under the auspices of CTL, was enriched by the contributions of the teaching and non-teaching staff, and particularly the PhD scholars. Their expertise and dedication significantly enhanced our learning experience. The PhD scholars' contributions were particularly noteworthy, as they imparted knowledge in Arduino IDE, IoT fundamentals tailored for IT, CSN and ES students, and conducted daily Python programming sessions. Their guidance played a pivotal role in our understanding and proficiency in the AIOT field.

We are deeply grateful to Dr. Kishore Kumar for providing us with the opportunity to acquire such valuable knowledge through this internship.

Special thanks are due to Boeing for their generous sponsorship, which was crucial to the success of this internship program at NIT Warangal .We also extend our sincere thanks to the faculty and staff of IITJ College and KITSW College for their continued support and encouragement throughout our academic journey.

In conclusion, we sincerely thank everyone involved for their invaluable support and contributions. This journey would not have been possible without you, and we are truly privileged to have been part of this enriching experience.

## ABSTRACT

Smart Luggage System is an advanced feature that can be useful for automatic driving, GPS Tracking and automatic weight detection. This will be useful for travellers to check and maintain their luggage, it is user-friendly travels along your mobile Bluetooth signals and mostly useful for people who are in oldage and to the passengers travelling in airlines.

## TABLE OF CONTENTS

**CHAPTER PAGE NO.**

COVER PAGE………………………………………………………… i

CERTIFICATE………………………………………………………… ii

ACKNOWLEDGEMENT……………………………………………... iii

[ABSTRACT……………………………………………………………. iv](#_TOC_250001)

TABLE OF CONTENTS………………………………………………. v

LIST OF FIGURES……………………………………………………. vi

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO** |
| 1 | INTRODUCTION   * 1. Background to the study   2. Problem statement   3. Aim of the study   4. Objectives of the study | 1-3 |
| 2 | LITERATURE SURVEY | 4- |
| 3 | METHODOLOGY   * 1. Hardware Description   2. Algorithm for overall setup   3. Position data transfer via Bluetooth module   3.4 Obstacle detection and automation algorithm | 6-32 |
| 4 | RESULTS | 33 |
| 5 | CONCLUSION | 34 |
| 6 | FUTURE SCOPE | 35-37 |
| 7 | REFERENCES | 38 |

|  |  |  |
| --- | --- | --- |
|  | **LIST OF FIGURES** |  |
| **FIGURE** | **DESCRIPTION OF FIGURE** | **PAGE No.** |
| 2.1.1. | Bluetooth Module, Ultrasonic sensor | 7 |
| 2.1.2 | 20kg Load cell & HX711 | 8 |
| 2.1.3 | Aurdino, Jumper wires | 9 |
| 2.1.4 | DC Motors and Motor driver | 12 |

## INTRODUCTION

Integrating the Internet of Things (IoT) in various applications has significantly enhanced communication and monitoring capabilities. One practical implementation is a Smart Luggage System that can be followed back a person by his device Bluetooth signals, combined with a Automatic weight detection and obstacle detection. This project utilizes the Bluetooth module for wireless connectivity, the load sensor for weighing and ultrasonic sensor for obstacle detection.

#### Background to the study

The rapid advancement in Internet of Things (IoT) technology has revolutionized various sectors by enabling efficient, real-time data collection and communication. IoT-based systems are increasingly being utilized for smart infrastructure, including Smart Luggage System. Normal luggage bags require manual physical efforts, which can be time-consuming and hardworking for oldage and physically disabled people. By integrating IoT with wireless technology, specifically using the Bluetooth module and load sensor, it is possible specifically using the Bluetooth module and load sensor, it is possible by including load sensor, Bluetooth module and ultrasonic sensor.

1

#### 1.2 Problem statement

Normal travelling luggage bags require more human for lifting and carrying and we do spend alot of money for luggage carrying persons, in some conditions it is a big task for aged and disabled people.

While travelling making courier of some luggage a person needs to always check the weights of the objects while keeping into the bag. In this weight checking is important.

Sometimes people may keep a bag at a distance and go to some personal works in such conditions we need to find the location of bag in searching of it.

By this project we can overcome such a kind of problems.

#### 2

#### 1.3 Aim of the study

The Smart Luggage System is a cost-effective, user-friendly travel solution designed for the elderly and children. It features automated phone following to reduce the burden of carrying bags, obstacle avoidance for safe navigation, and sensor-based weight detection to prevent excess baggage fees. This system aims to make travel easier, safer, and more comfortable.

**1.4 Objective of the study**

The Smart Luggage System project aims to enable automatic following, implement obstacle detection, incorporate weight measurement, ensure a user-friendly interface, and maintain cost-effectiveness. It enables luggage to follow the user's mobile phone, enhances safety with obstacle detection, provides real-time weight feedback, ensures easy operation, and remains affordable for a wide range of users.

3

## 2.LITERATURE SURVEY

**1.Luggage Tracking System Using IoT**

This project uses a GPS module integrated with Arduino and an alarm sensor to continuously track the location of the luggage. The system alerts users if the bag moves beyond certain predefined distances (10m, 20m, and 30m) to prevent theft or loss.

**2.Smart Luggage Tracker Using RFID**

The system employs RFID technology for tracking and handling luggage. RFID tags, which contain microchip data, are attached to luggage. These tags can be read remotely by RFID readers, providing detailed information about the luggage's location, status, and handling at various checkpoints.

**3.Smart Luggage Tracking and Alert System Using Arduino**

This system features advanced security technologies such as fingerprint sensors and cloud-based storage for authorised and unauthorised location details. It includes GSM-GPS modules for tracking and sending location data to the user’s smartphone.

4

**2.1.Gaps Identified**

### Bluetooth Proximity Range:

* Range Definition: Establish the required range so that the Bluetooth module can track the user's phone efficiently.
* Signal Strength: Verify that there is enough Bluetooth signal strength to keep the user's phone connected steadily.
* Handling Interference: Take into account possible interference sources (such as other Bluetooth devices) and lessen their impact on the system.

### Obstacle Detection:

* Accuracy of Sensor: To successfully detect obstacles, make sure the ultrasonic sensor offers accurate distance measurements.
* Detection Range: Establish the ideal range of detection that the ultrasonic sensor should have to identify any obstructions in the bag's route.
* Integration: Make sure the ultrasonic sensor has a clear line of sight for obstacle detection and include it into the bag's design.

### Weight Detection:

* Sensor Calibration: Calibrate the weight detection sensor to accurately measure the weight of the bag.
* Weight Limit: Define the weight limit that the sensor can accurately detect to ensure it meets the requirements of the bag.
* Integration: Integrate the weight detection sensor into the bag's design without affecting its functionality or comfort for the user.

## 5

## Design Methodology:

### Hardware Setup:

1. **Bluetooth Module Connection:**

* Attach the VCC pin of the Bluetooth module to the 5V pin of the Arduino.
* Attach the GND pin of the Arduino to the GND pin of the Bluetooth module.
* Attach pin 2 of the Arduino to the TX pin of the Bluetooth Module.
* Attach pin 3 of the Arduino to the RX pin of the Bluetooth Module.

1. **Ultrasonic Sensor Connection:**

* Attach the 5V pin of the Arduino to the VCC pin of the ultrasonic sensor.
* Attach the GND pin of the ultrasonic sensor to the GND pin of the Arduino.
* Attach digital pin 12 of the Arduino to the TRIG pin of the ultrasonic sensor.
* Attach digital pin 11 of the Arduino to the ECHO pin of the ultrasonic sensor.

1. **HX711 Weight Sensor Module to the Arduino:**

* Attach digital pins 6, 7, 8 and 9 of Arduino to the IN1, IN2, IN3, and IN4 pins of the L298N module, accordingly.
* Attach the left motor to the L298N module's OUT1 and OUT2 ports.
* Attach the correct motor to the L298N module's OUT3 and OUT4.

1. **Load Cell to the HX711 Weight Sensor Module:**

* Connect E+ of the module to the red wire of the load cell.
* Connect E- of the module to the black wire of the load cell.
* Connect A+ of the module to the green wire of the load cell.
* Connect A- of the module to the white wire of the load cell.

6

1. **Power Supply:**

* Attach the 3.7V lithium-ion battery's positive connection to the 12V pin of the motor driver.
* Attach the battery's negative connection to the GND pin of motor driver as well as the Arduino's GND Pin

1. **Additional Components:**

* Attach the DC motors to the motor wheels.
* Install the weight and ultrasonic sensors on the bag in the proper locations.

**7.Overall Setup:**

* Inside the bag, firmly attach the Arduino, L298N motor driver, Bluetooth module, and sensors.
* Verify that every connection is safe and that the parts are positioned correctly for best performance

## Automation:

Objective: To allow the bag to follow the user's phone without human assistance, implement automation.

Implementation:

* Determine the desired direction and distance to the phone using GPS data that you obtained from MATLAB.
* Create an algorithm that modifies the bag's wheel movement in real time to track the phone.

Testing:

* Experiment with the automation feature in various situations, such strolling, reversing course, and halting suddenly.

7

* Verify that the bag reacts precisely to adjustments in the phone's location

**Obstacle Detection:**

1. Objective:

Give the bag the ability to sense objects in its path so that collisions are avoided.

1. Implementation:

* Use the ultrasonic sensor in front of the bag to identify obstacles that are within a specific range.
* When an obstruction is discovered, incorporate the obstacle detection algorithm into the main software to stop the bag or alter its trajectory.

1. Testing:

* Try out the obstacle detection function by putting different sized and kind of objects in front of the bag.
* Make sure the bag stops or changes course when it encounters an obstruction.

8

## Weight Sensing:

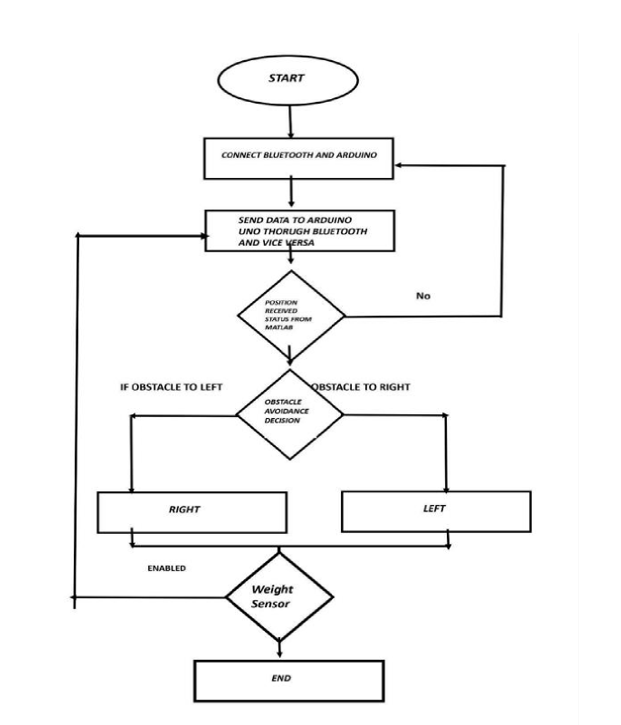
1. Objective:

* Give the bag permission to determine its own weight so that it can inform the user.

1. Implementation:
   * Determine the bag's weight using the weight sensor.
   * Add code to the main software that will notify or show the user if the weight over a predetermined threshold.
2. Testing:
   * Adjust the bag's weight to test the weight detecting feature.
   * Make sure the bag weighs itself accurately and gives the user the information information they need.

9

**BLOCK DIAGRAM**

****

10

# ALGORITHM FOR OVERALL SETUP:

Initialize Components:

* Set up the L298N motor driver, weight sensor, ultrasonic sensor, Arduino Uno, Bluetooth module, and other required parts.

Setup Bluetooth Communication:

* Configure the Bluetooth module to establish communication with the user's phone.

Setup Ultrasonic Sensor:

* Set up the ultrasonic sensor so that it can identify objects in front of the Bag.

Setup Motor Control:

* Set up the L298N motor driver to regulate the DC motors' movement.

Main Loop:

* Create a loop to carry out the subsequent actions repeatedly.

Receive Position Data:

* To find the position of the phone, use Bluetooth to receive position data from MATLAB.

Calculate Movement:

* Determine the direction and distance to the phone using the received Position data.
* Modify the bag's wheel movement so that it follows the phone.

Obstacle Detection:

* Check for obstructions in front of the bag using the ultrasonic sensor.
* If an obstacle is found, halt the bag or adjust its course to avoid a Collision.

11

Weight Sensing:

* Measure the bag's weight with the weight sensor.
* If the weight is more than a predetermined amount, notify the user.

Repeat:

* To continually update the bag's movement based on obstacle detection, weight sensing, and phone position, repeat the main loop

End:

* When the user manually stops the algorithm, or when the bag is no longer in use, it ends.

**Position Data Transfer via Bluetooth from MATLAB to Arduino:**

### Algorithm:

1. Initialize Mobile Device Object:

* Remove any object that is currently on a mobile device
* Set up the object m for the mobile device.

clear mobiledev; clear m;

m = mobiledev;

Enable Logging:

To enable location data logging, set m.Logging to 1.

m.Logging 1;

Pause for Data Collection:

* + Give MATLAB position sensor data a few seconds to gather before continuing.

pause(5)

12

Continuous Data Transfer Loop:

* Create a loop to obtain and show MATLAB position sensor data continually.
* To obtain the most recent MATLAB position sensor data, using the poslog function.

while true [latitude, longitude, altitude, speed, heading, timestamp] = poslog(m);

check for valid data:

* Verify that the data for latitude and longitude are not empty.
* Show the MATLAB location sensor data if it is valid.
* If not, show a notice saying that we are waiting for valid MATLAB position sensor data.

if ~isempty(latitude) && ~isempty(longitude) fprintf('Latitude: %.6f, Longitude: %.6f, Altitude: %.6f, Speed: %.6f, Heading: %.6f, Timestamp: %.6f\n', latitude(end), longitude(end), altitude(end), speed(end), heading(end), timestamp(end)); else fprintf('Waiting for valid MATLAB position sensor data...\n'); end

Pause Before Next Reading:

* Take a little break before to the subsequent reading in order to prevent constant and swift data transfer.

pause(2); end

End of Loop:

* To continually update and upload MATLAB position sensor data, repeat the loop forever.

13

## Obstacle Detection and Automation Algorithm:

### Algorithm:

1. Initialization:

* Intiate serial connection with the SIM808 module.
* Establish the button pin for the action that the user initiates.
* Configure the phone number that will receive the SMS.

1. Setup:

* Start serial communication with the SIM808 module and for debugging.
* Turn on GPS power and choose the RMC NMEA sentence to obtain location information.

1. Main Loop:

* Keep checking the button for feedback from the user.
* When the button is pressed, the subsequent actions take Place:
* Get location sensor data in MATLAB.
* Put an obstacle detection algorithm into practice.
* Manage the bag's movement according to the obstacle identification.

**Obstacle Detection Algorithm:**

* Make use of the ultrasonic sensor to identify any obstructions ahead of the bag.
* Stop the bag or alter its course to avoid obstacles if one is identified within a specific range
  + nection with the SIM808 module, initialize the SoftwareSerial library.
    - Establish the button pin for the action that the user initiates.
    - Configure the phone number that will receive the SMS.

1. Setup:
   * Start serial communication with the SIM808 module and for debugging.
   * Turn on GPS power and choose the RMC NMEA sentence to obtain location information.
2. Main Loop:
   * Keep checking the button for feedback from the user.
     + When the button is pressed, the subsequent actions take Place:
       - Get location sensor data in MATLAB.
       - Put an obstacle detection algorithm into practice.
       - Manage the bag's movement according to the obstacle identification.
3. Obstacle Detection Algorithm:
   * Make use of the ultrasonic sensor to identify any obstructions ahead of the bag.
   * Stop the bag or alter its course to avoid obstacles if one is identified within a specific range.

* Sensor data from MATLAB.
* Move the bag to the appropriate location if no obstacles are found.
* Stop the sensor bag or adjust its direction if an obstruction is detected.

14

### Weight Sensing Algorithm:

**Algorithm:**

1. Initialization:

* Add the libraries required for the LiquidCrystal I2C and the HX711 ADC.
* Set the LiquidCrystal I2C for the LCD and the HX711 ADC for the load cell to their initial values.

1. Setup:

* Start the HX711 ADC connection and give it some time to stabilize.
* Based on your configuration, set the load cell's calibration factor.
* Set the backlight and LCD to initial settings.

1. Main Loop:

* Update and download data from the load cell continuously.
* Show the LCD's weight in grams

15

**Merits of the IoT-Based Smart Luggage System:**

1. **Convenience**: Enables easier transportation of heavy luggage through automated movement.
2. **Security**: Incorporates GPS tracking for real-time location updates.
3. **Cost-Effective**: Designed as a low-cost, user-friendly system.
4. **Control Features**: Allows users to start and stop the bag using a smartphone app.

**Demerits:**

1. **Dependency on Technology**: Requires a stable smartphone connection for optimal functionality.
2. **Battery Limitations**: Device performance relies on consistent power supply.
3. **Potential Durability Issues**: Components may wear out under heavy usage.

16

**EXPERIMENTATION AND RESULTS:**

### Matlab Online-

% Clear any existing mobile device object clear mobiledev;

clear m;

% Initialize mobile device object m = mobiledev;

% Enable logging of location data m.Logging = 1;

% Define the file path

filePath = '/MATLAB Drive/GPSData.mat';

% Delete the existing file if it exists if exist(filePath, 'file') == 2

delete(filePath); end

% Loop to continuously log GPS data while true

% Get the current GPS data latitude = m.Latitude; longitude = m.Longitude;

% Check if the data is valid

if ~isempty(latitude) && ~isempty(longitude) && ~any(isnan(latitude)) &&

~any(isnan(longitude))

% Save the last valid latitude and longitude

Lat=

17

latitude(end); lon = longitude(end);

% Display the GPS data

fprintf('Latitude: %.6f, Longitude: %.6f\n', lat, lon);

% Save to MATLAB Drive save(filePath, 'lat', 'lon');

else

fprintf('Waiting for valid GPS data...\n'); end

% Pause for a short while before the next reading pause(2);

end

### Downloaded Matlab:

% Clear existing Bluetooth object clear bt;

% Create Bluetooth object (replace 'HC-05' with your Bluetooth module's name) bt = bluetooth('HC-05', 1);

while true

% Check if the GPS data file exists if exist('GPSData.mat', 'file')

try

% Load the GPS data load('GPSData.mat', 'lat', 'lon');

catch ME

fprintf('Error: Unable to load GPSData.mat. %s\n', ME.message); pause(2);

18

continue; end

% Create the command string

command = sprintf('MOVE,%.6f,%.6f\n', lat, lon);

% Send the command to the microcontroller write(bt, command, "string");

% Display the command for debugging fprintf('Sent: %s\n', command);

else

% Display a message if the file does not exist fprintf('Error: GPSData.mat file not found.\n');

end

% Pause for a short while before the next reading pause(2);

end

### Arduino

#include <TinyGPSPlus.h>

#include <SoftwareSerial.h>

#include <Wire.h>

#include "HX711.h"

// Bluetooth setup

SoftwareSerial BTSerial(2, 3); // RX, TX

// Motor Driver Pins

const int motorPin1 = 9; // Left motor forward

const int motorPin2 = 8; // Left motor backward

const int motorPin3 = 7; // Right motor forward

const int motorPin4 = 6; // Right motor backward

// Ultrasonic Sensor Pins

19

const int trigPin = 12;

const int echoPin = 11;

// Load Cell Pins

const int LOADCELL\_DOUT\_PIN = 4;

const int LOADCELL\_SCK\_PIN = 5;

// Initialize the HX711

HX711 scale;

// TinyGPS++ object

TinyGPSPlus gps;

// Target GPS coordinates

float targetLat = 0.0;

float targetLon = 0.0;

// Flag to indicate if GPS coordinates are received

bool gpsReceived = false;

// Timer to check for stale coordinates

unsigned long lastCoordTime = 0;

const unsigned long COORD\_TIMEOUT = 10000; // 10 seconds

void setup() {

// Serial communication setup for debugging

Serial.begin(9600);

BTSerial.begin(9600);

// Motor Driver Pins as Outputs

pinMode(motorPin1, OUTPUT);

pinMode(motorPin2, OUTPUT);

pinMode(motorPin3, OUTPUT);

pinMode(motorPin4, OUTPUT);

// Ensure motors are stopped at startup

stopMotors();

// Ultrasonic Sensor Pins

pinMode(trigPin, OUTPUT);

pinMode(echoPin, INPUT);

// Initialize load cell

scale.begin(LOADCELL\_DOUT\_PIN, LOADCELL\_SCK\_PIN);

Serial.println("System Initialized. Waiting for data...");

}

void loop() {

// Bluetooth data handling

if (BTSerial.available()) {

String data = BTSerial.readStringUntil('\n');

20

Serial.print("Received via Bluetooth: ");

Serial.println(data); // Print received data for debugging

// Check if the received data starts with "MOVE"

if (data.startsWith("MOVE")) {

// Extract the latitude and longitude

int firstCommaIndex = data.indexOf(',');

int secondCommaIndex = data.indexOf(',', firstCommaIndex + 1);

if (firstCommaIndex > 0 && secondCommaIndex > 0) {

String latString = data.substring(firstCommaIndex + 1, secondCommaIndex);

String lonString = data.substring(secondCommaIndex + 1);

// Convert to float

targetLat = latString.toFloat();

targetLon = lonString.toFloat();

// Print for debugging

Serial.print("Target Latitude: ");

Serial.println(targetLat, 6); // Print with 6 decimal places for accuracy

Serial.print("Target Longitude: ");

Serial.println(targetLon, 6); // Print with 6 decimal places for accuracy

// Set flag to indicate GPS coordinates received

gpsReceived = true;

lastCoordTime = millis(); // Reset the timer

}

}

}

// If GPS coordinates have not been received or are stale, stop motors

if (!gpsReceived || millis() - lastCoordTime > COORD\_TIMEOUT) {

stopMotors();

if (gpsReceived) {

Serial.println("GPS coordinates stale, stopping motors.");

gpsReceived = false; // Reset the flag

}

return; // Skip the rest of the loop if no valid coordinates

}

// Get distance from ultrasonic sensor

long distance = getDistance();

// Obstacle detection

if (distance <= 20) {

stopMotors();

delay(100);

moveBackward();

delay(500);

stopMotors();

delay(100);

turnLeft();

delay(500);

stopMotors();

} else {

// Calculate direction and move towards target

moveToTarget();

}

// Get weight from load cell

float weight = scale.get\_units(5); // Get the average of 5 readings

Serial.print("Weight: ");

Serial.print(weight);

Serial.println(" kg");

delay(100); // Short delay to avoid excessive serial output

}

long getDistance() {

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

long duration = pulseIn(echoPin, HIGH);

long distance = duration \* 0.034 / 2;

return distance;

}

void moveToTarget() {

// Use TinyGPS++ to calculate distance and course to target

float distanceToTarget = TinyGPSPlus::distanceBetween(

gps.location.lat(),

gps.location.lng(),

targetLat,

targetLon

);

float courseToTarget = TinyGPSPlus::courseTo(

gps.location.lat(),

gps.location.lng(),

targetLat,

targetLon

);

Serial.print("Distance to Target: ");

Serial.println(distanceToTarget);

Serial.print("Course to Target: ");

Serial.println(courseToTarget);

// Move towards target based on calculated course

// Move towards target based on calculated course

if (distanceToTarget > 5.0) { // Threshold distance to avoid oscillations

if (courseToTarget >= 0 && courseToTarget < 180) {

turnRight();

} else {

turnLeft();

}

moveForward();

} else {

stopMotors();

Serial.println("Reached Target Coordinates.");

gpsReceived = false; // Reset the flag once target is reached

}

}

void moveForward() {

Serial.println("Moving Forward");

analogWrite(motorPin1, 120); // Adjust speed as needed

digitalWrite(motorPin2, 0); // Left motor backward (off)

analogWrite(motorPin3, 10); // Adjust speed as needed

digitalWrite(motorPin4, 0); // Right motor backward (off)

}

void moveBackward() {

Serial.println("Moving Backward");

digitalWrite(motorPin1, 0);

analogWrite(motorPin2, 120); // Adjust speed as needed

digitalWrite(motorPin3, 0);

analogWrite(motorPin4, 10); // Adjust speed as needed

}

void turnRight() {

Serial.println("Turning Right");

digitalWrite(motorPin1, 120);

digitalWrite(motorPin2, 0);

digitalWrite(motorPin3, 0);

digitalWrite(motorPin4, 10);

}

void turnLeft() {

Serial.println("Turning Left");

digitalWrite(motorPin1, 0);

digitalWrite(motorPin2, 120);

digitalWrite(motorPin3, 10);

digitalWrite(motorPin4, 0);

}

void stopMotors() {

Serial.println("Stopping Motors");

digitalWrite(motorPin1, 0);

digitalWrite(motorPin2, 0);

digitalWrite(motorPin3, -100);

digitalWrite(motorPin4, -100);

}

# RESULTS:

Fig:1 Live Data Transmission

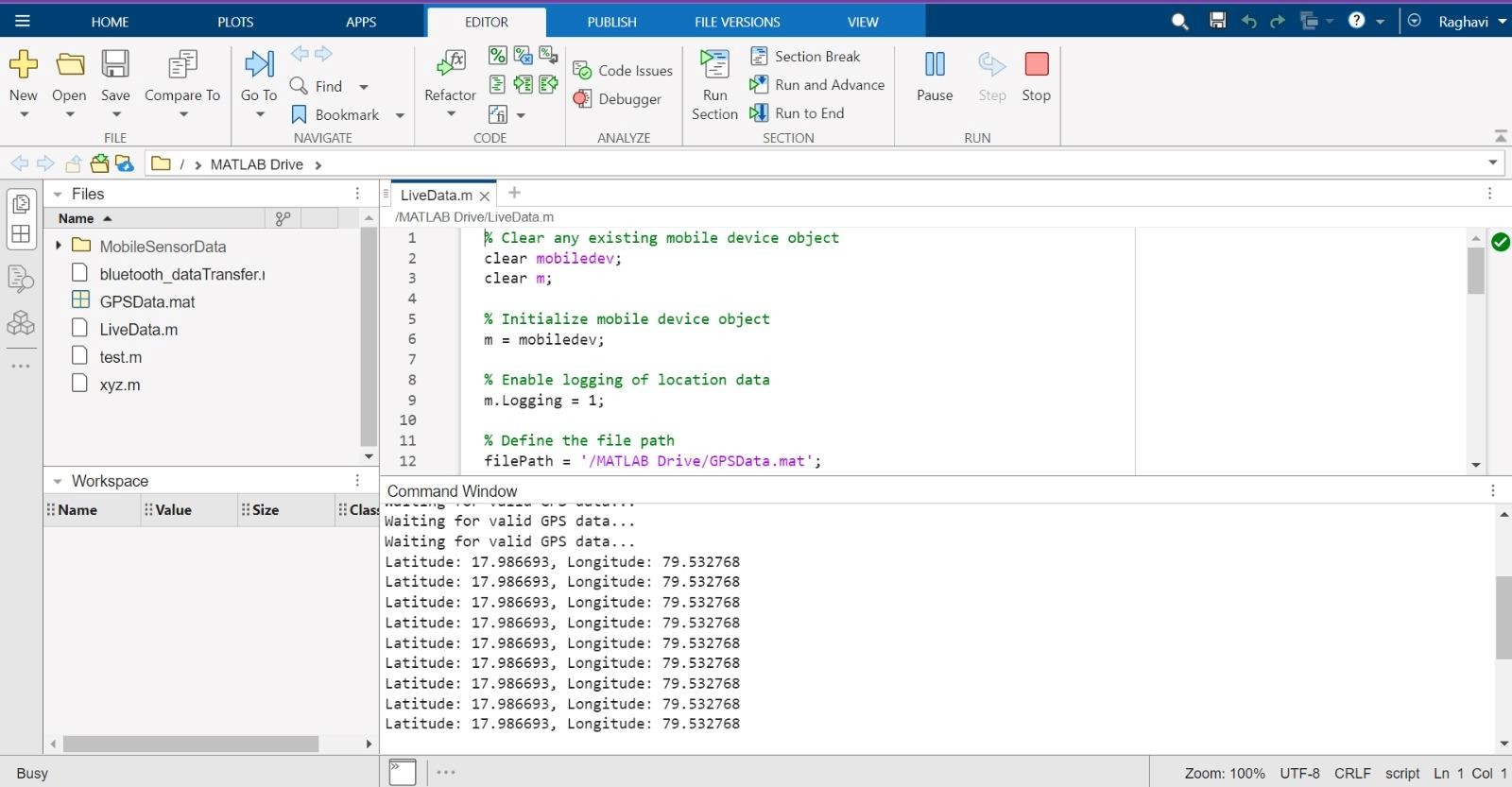


Fig2: Live Data Transmitted to Bluetooth Module

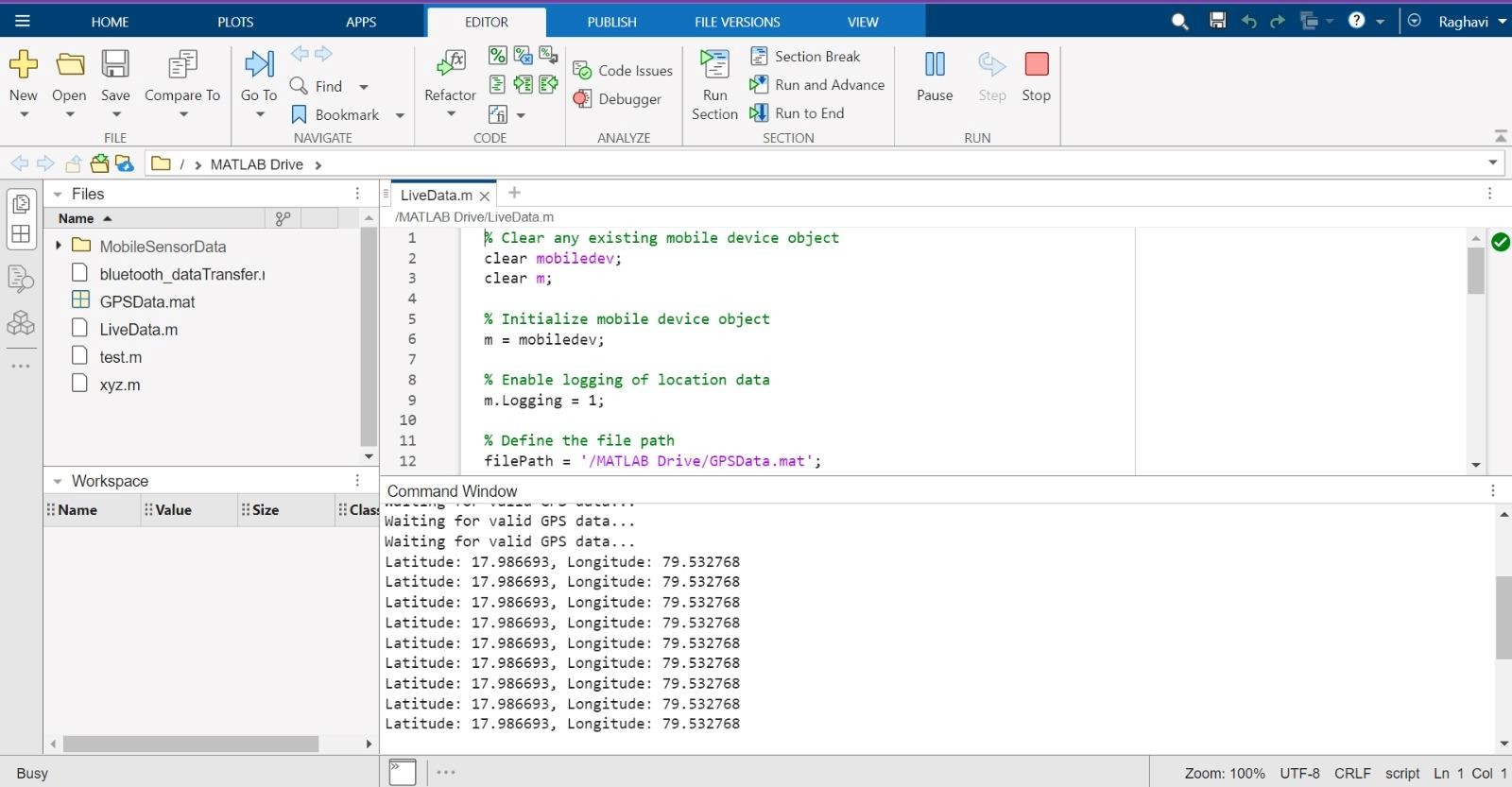


Fig3: Weight Detection in Serial Monitor of Arduino IDE

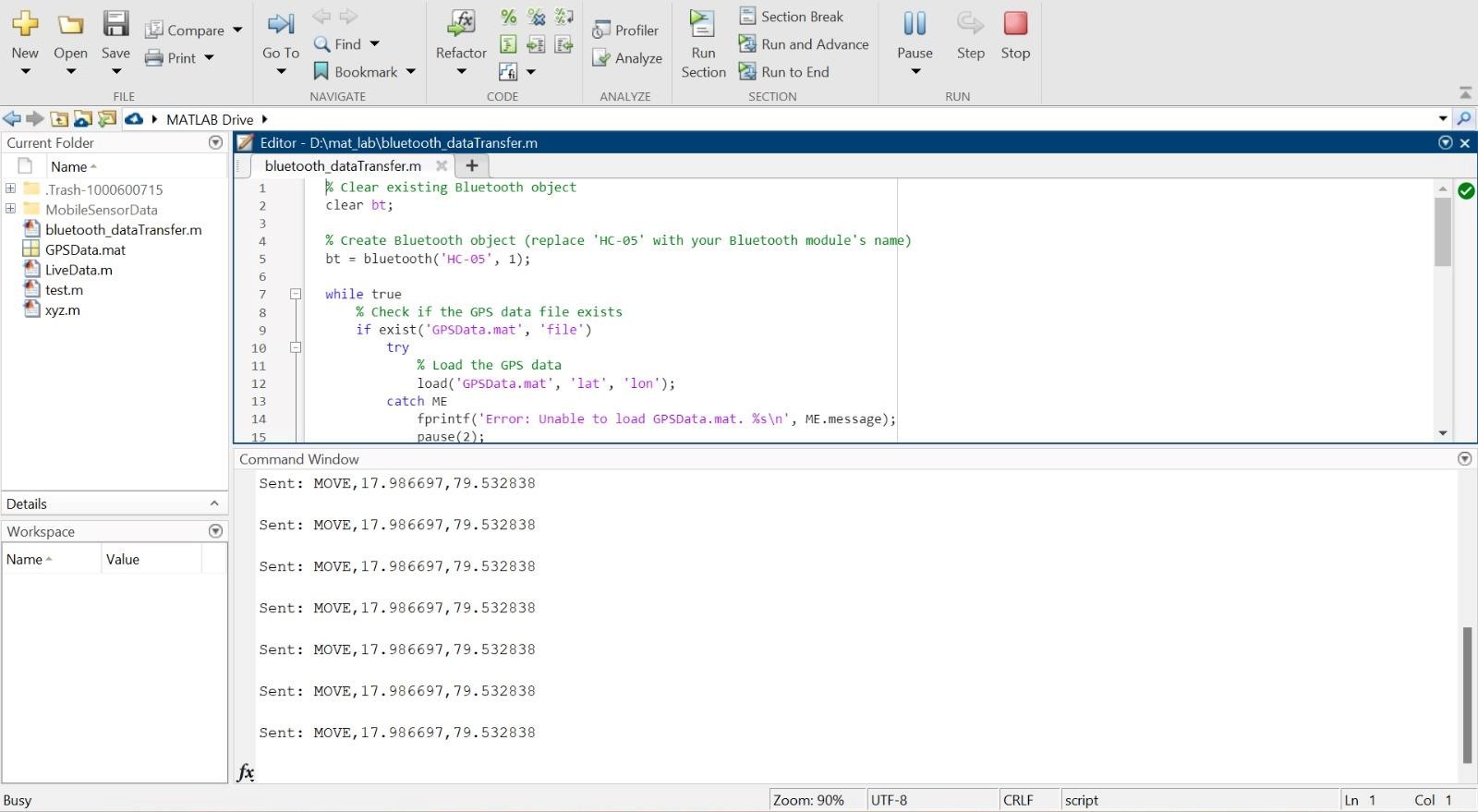
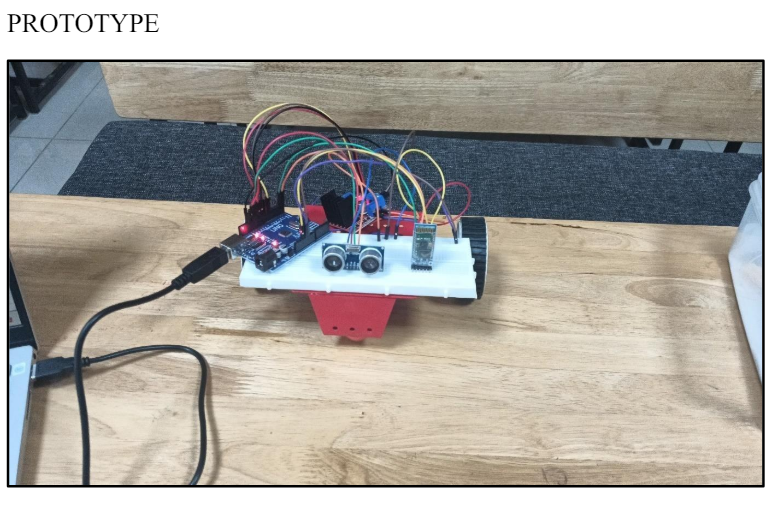


Fig:4 Prototype



**CONCLUSION:**

In conclusion, the Smart Traveler Bag project combines a number of ideas and technology to improve the travel experience, particularly for young people and the elderly. The automated component of the bag ensures simplicity and ease of use by following the user's phone automatically, using MATLAB location sensor data. Using an ultrasonic sensor, the obstacle detection feature gives the bag an extra degree of security by allowing it to recognize and steer clear of anything in its route. This feature is essential, particularly in places that are crowded or unfamiliar and may have restricted visibility. Furthermore, the weight sensing feature gives customers access to real-time weight information about the bag by using a load cell and a HX711 ADC.This is especially helpful for travelers who have weight requirements to follow, including airline weight limitations.In general, the Smart Traveler Bag project demonstrates how to combine sensor technologies, software algorithms, and hardware parts to provide a clever and convenient travel item. It illustrates how IoT and smart gadgets may improve daily life and address typical travel-related problems.

# FUTURE SCOPE:

To enhance the Smart Luggage System, integrating a Smart Digital Lock for improved security and a GPS Tracking System for real-time location monitoring and retrieval in case of loss would be beneficial. Additionally, implementing a user-friendly interface for easy interaction and ensuring cost-effectiveness would further enhance its appeal and utility.

**REFERENCES:**

IJRASET- International Journal For Research in Applied Science and Engineering Technology

Title- IoT Based Smart Bag using Arduino

Volume- 10

Issue- X|

Month of Publication- November 2022

DOI- https://doi.org/10.22214/ijraset.2022.47615

https://www.ijraset.com/research-paper/iot-based-smart-bag-using-arduino

https://www.ijraset.com/best-journal/iotbased-smart-bag-using-arduino

https://www.ijert.org/smart-travel-bag

.google.com/patent/US20170220040A1/en

https://patents.google.com/patent/US20170049202A1/en