UNIVERSITY OF VOCATIONAL TECHNOLOGY

Faculty of Engineering Technology

Department of Electro-Mechanical Technology

EE402040

Internet of Things (IoT)

IoT-Based Car Parking Assistance System

Project Report

Group Members

Name	**Reg. no**
G.G.H.D.Fernando	: MEC/22/B1/12
U.M.G.G.S.D.Udakumbura	: MEC/22/B1/18

Program	: B.Tech in Mechatronics Technology
Submission Date	: 31st July 2025

Instructed by: Mr. Janith Kasun

Contents

IoT-Based Car Parking Assistance System

1. Introduction

Parking maneuver safety is a persistent problem, especially in crowded or confined environments. The project introduces an affordable, real-time IoT-based car parking assistance system leveraging ESP32 microcontrollers and ultrasonic sensors, with intuitive visual and audible feedback via mobile and web dashboards. The design emphasizes modularity, reliability, and wide vehicle compatibility, aiming to bridge the gap between basic sensors and expensive commercial systems

2. Objectives

The Car Parking Assistant System is an IoT-based device that provides real-time assistance to drivers by detecting obstacles in the front, rear, left, and right zones of the vehicle. It transmits sensor data wirelessly to a central ESP32 controller, which processes the information and relays it visually and audibly to the user. The system improves parking safety and awareness using both remote mobile and local dashboard interfaces.

Key Objectives:

- Measure distance from all four surrounding zones of the vehicle with ultrasonic sensors for complete perimeter coverage.
- Send sensor data wirelessly using the ESP-NOW protocol for low-latency, reliable communication between sensor nodes and the main controller.
- Provide live zone status and alerts on a Blynk-based mobile dashboard for convenient remote monitoring and feedback.
- Host a custom HTTP-based web dashboard served by the main ESP32, accessible via a local WiFi IP (e.g., 192.168.8.103), allowing in-car or PC display of real-time parking data even without an internet connection.

3. Components

Component	Qty.	Justify	Picture
ESP32	5	• Dual-core MCU with built-in Wi-Fi/Bluetooth • Controls sensors and handles wireless communication	
Ultrasonic sensor (HC-SR04 Sensor)	4	• Measures distance using sound waves • Range: 2 cm to 400 cm	
Buzzer	1	• Emits beeping sounds for danger alerts • Beeping frequency increases as the obstacle nears	
Android TV	1	• Mounted on the dashboard to act as a live interface • Shows top-down view with visual distance alerts	
5V battery	5	• Power for each ESP32 (e.g., power bank or USB)	

3.1. Bill of Materials

Item	Quantity	Unit Price (Rs.)	Total Cost (Rs.)
ESP32 Microcontroller	5	1,340	6,700
HC-SR04 Ultrasonic Sensor	4	350	1,400
Breadboard + Jumper Wires	one set	800	800
5V Battery	5	500	2,500
Miscellaneous (tape, glue)	-	500	500
Mobile device for testing	1	-	personal use
Laptop for programming	2	-	personal use

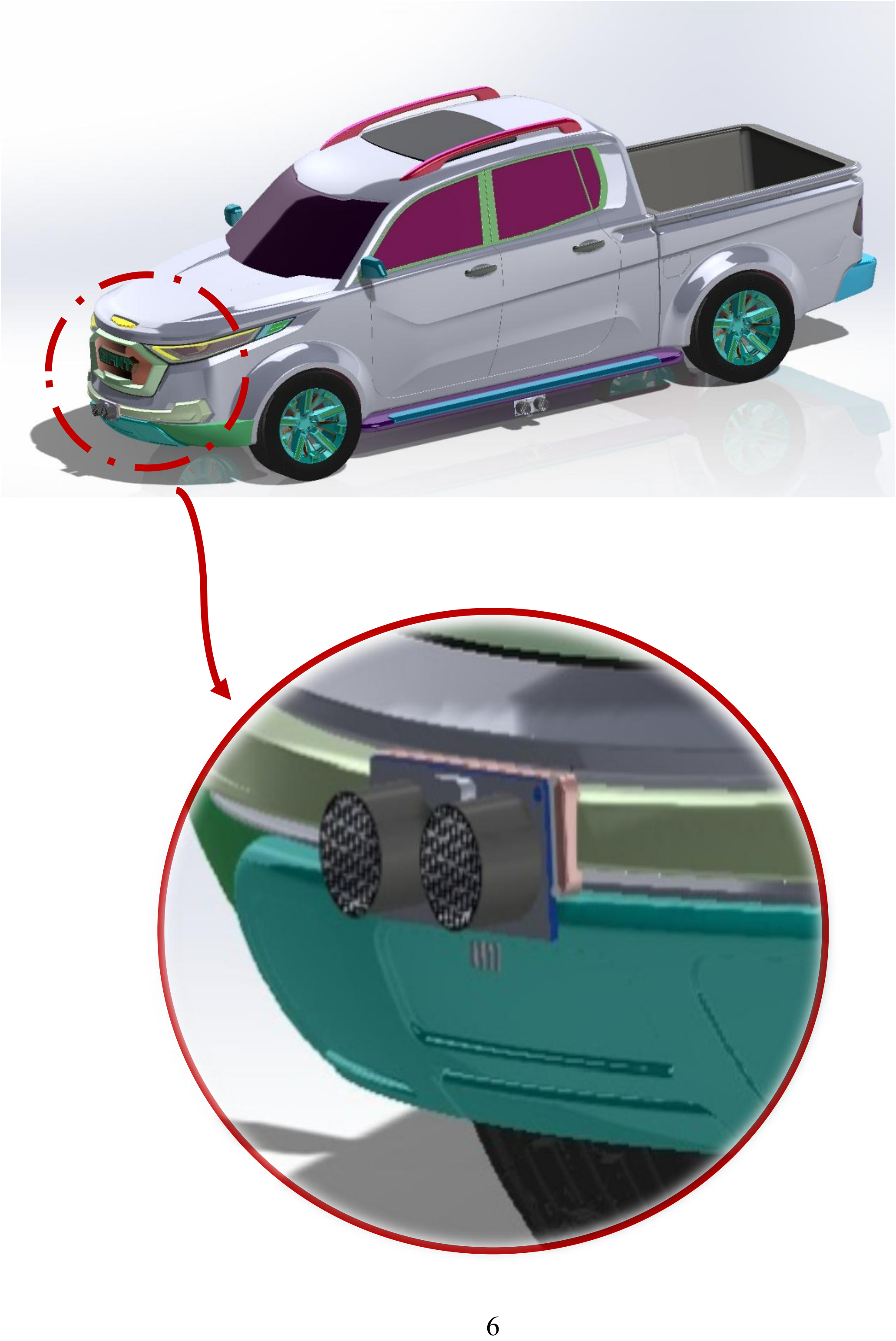
Estimated Total: Rs. 11,900.00

4. System Architecture & Methodology

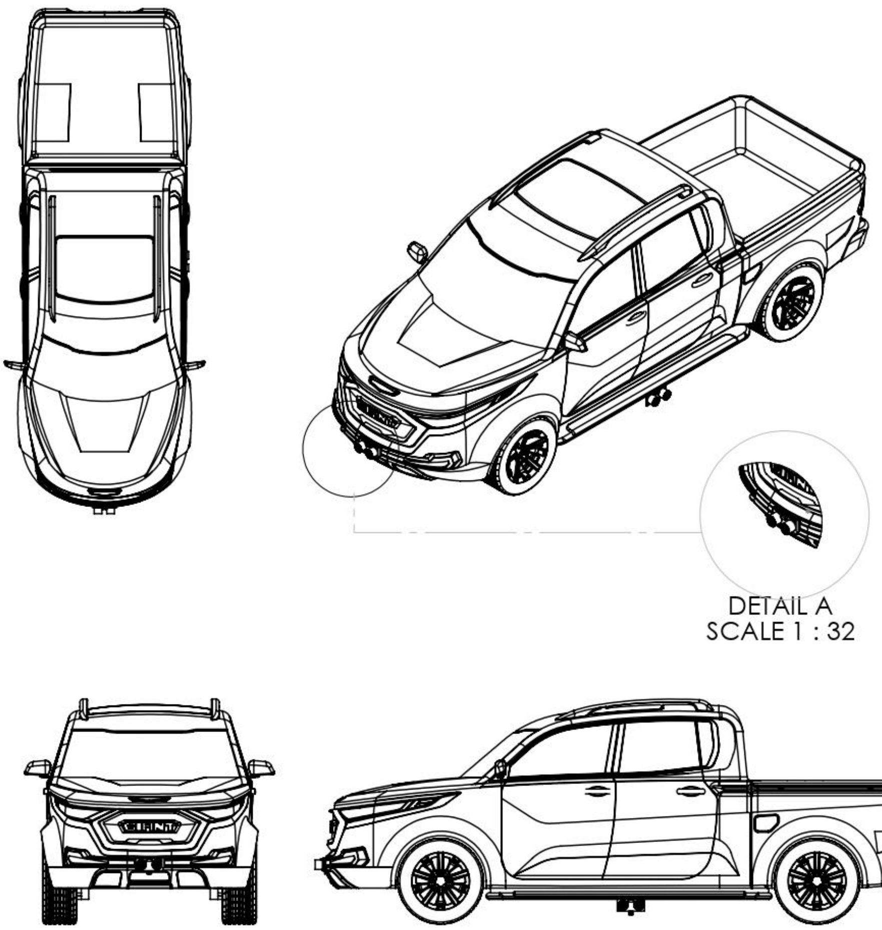
Physical Topology

- Four ESP32 sensor nodes (each with one ultrasonic sensor, oriented front, rear, left, right).
- One main ESP32 controller, connected to a buzzer and interfacing with dashboards.

4.1. Architecture



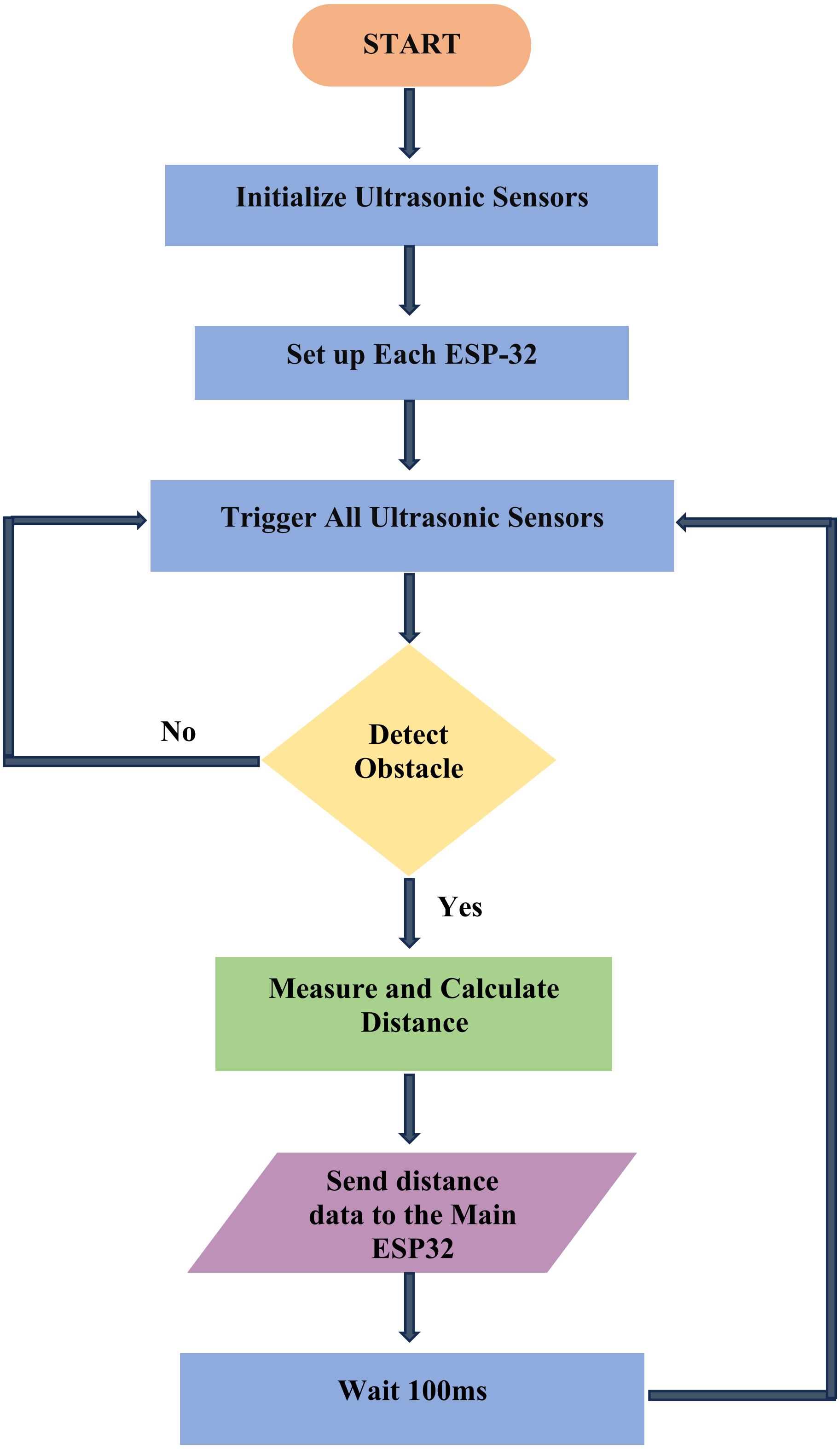
4.2. SolidWorks Design



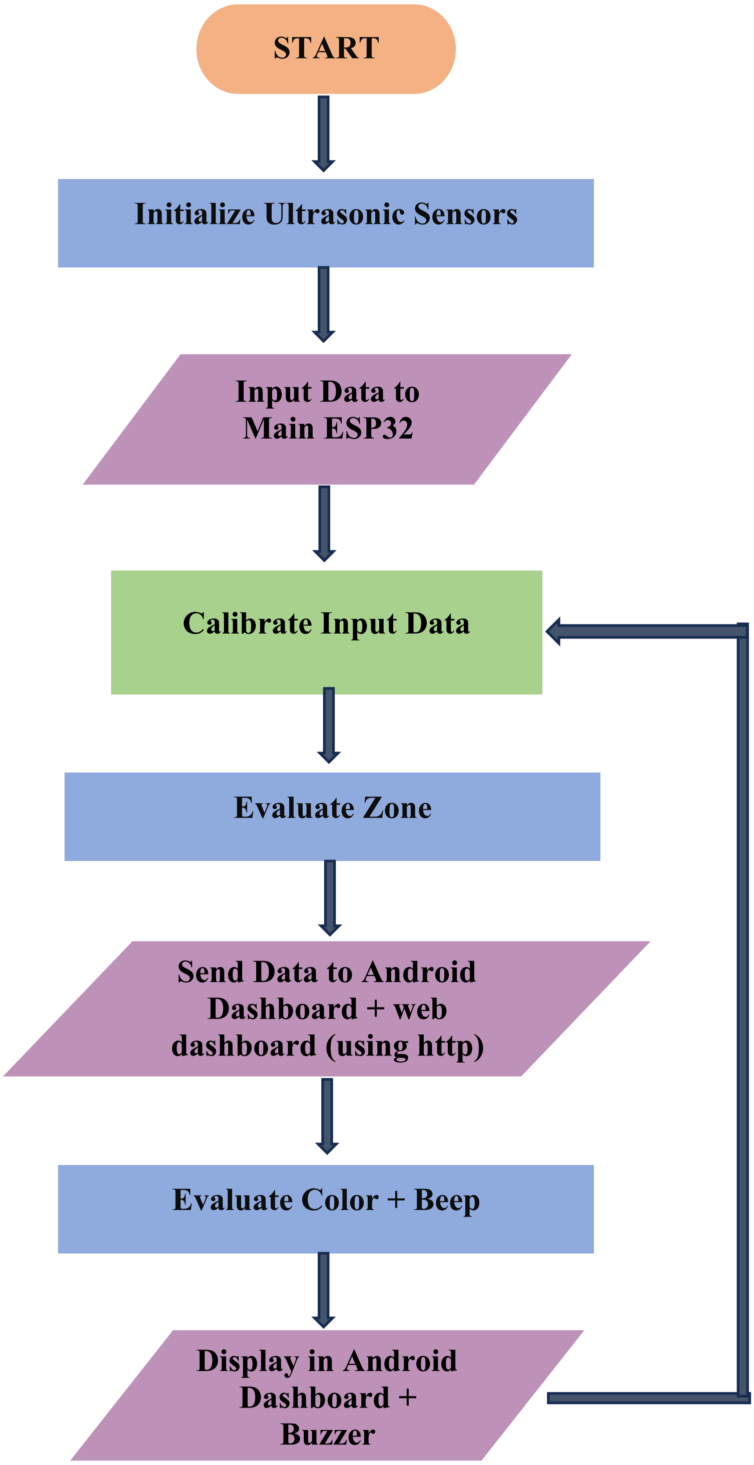
UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ANGULAR:	FINISH:	DEBURR AND BREAK SHARP EDGES	DO NOT SCALE DRAWING	REVISION

	NAME	SIGNATURE	DATE				
DRAWN							TITLE: Smart Car Assistant System
CHK'D							
APPV'D							
MFG							
Q.A				MATERIAL:			DWG NO. A4
				WEIGHT:			SCALE:1:64 SHEET 1 OF 1

4.3. Software


START
Initialize Ultrasonic Sensors
Set up Each ESP-32
Trigger All Ultrasonic Sensors
No
Detect Obstacle
Yes
Measure and Calculate Distance
Send distance data to the Main ESP32
Wait 100ms

This flowchart shows how an ESP32 sensor node works in a parking assistant system. It initializes sensors and sets up ESP-NOW communication, then continuously measures distance and sends the data with an ID to the central ESP32 for real-time monitoring.


START
Initialize Ultrasonic Sensors
Input Data to Main ESP32
Calibrate Input Data
Evaluate Zone
Send Data to Android Dashboard + web dashboard (using http)
Evaluate Color + Beep
Display in Android Dashboard + Buzzer

Software Overview

- Sensor node software: Distance measurement, wireless transmission.
- Main ESP32: Data collection, zone evaluation, buzzer control, data relay.
- Dashboards:
    - **Mobile (Blynk):** Real-time status using Blynk Cloud.
    - **HTTP (HTML dashboard):** Real-time data via HTTP (/status endpoint) served directly at 192.168.8.103.

5. How It Works

- Each sensor node ESP32 continuously measures distance to an obstacle via its assigned ultrasonic sensor, acting as a distributed detection layer for its car zone.
- Sensor nodes transmit distance values (with a zone ID) wirelessly to the main ESP32 using ESP-NOW, which does not require internet or a router.
- The main ESP32 controller receives all four zones' data, computes color-coded zone status (green, yellow, red) based on preset safety thresholds, and operates a buzzer for immediate collision risk alerts.
- **Dual dashboard output:**
  - **Mobile/Web via Blynk:** Main controller forwards zone distances and statuses to Blynk Cloud. Mobile app widgets update in real time to display numeric and color-coded alerts.
  - **HTTP/HTML Dashboard:** Main controller also acts as a web server on the local WiFi network (192.168.8.103), serving a webpage that presents the same zone status with a graphical plan view of the car.

6. Challenges Faced

- **ESP-NOW Communication Stability:**
  Synchronizing multiple ESP32 sensor nodes to reliably transmit to the main controller presented issues such as MAC address mismatches and inconsistent WiFi channels. These were overcome through careful address assignment, channel alignment, and code-level debugging.

- **Power Supply Glitches:**
  Maintaining continuous, stable power to each ESP32 (especially via power banks) was essential to prevent random resets and communication drops—resolved by quality testing of power sources and connection reliability.

- **Accurate Distance Detection in Real Scenarios:**
  Sensor readings were affected by mounting position, sensor orientation, and environmental interference. Systematic physical adjustment and real-world scenario testing ensured repeatable and reliable operation.

- **Dashboard Integration:**
  Ensuring real-time data and color logic responded accurately in both the mobile (Blynk) and local web dashboards entailed careful alignment of virtual pins, data endpoints, and real-time software refresh logic.

7. Possible Improvements

- **Battery Status Monitoring:**
  Integrate analog sensing and dashboard display of the 5V power/battery percentage for each sensor node and the controller, so users can proactively recharge and avoid unexpected shutdowns.
- **Enhanced Dashboard Features:**
  - Continue displaying distance readings (cm) for each zone in real time on the Blynk app.
  - Enhance the HTTP/HTML dashboard with a graphical plan view and animated/colored zone alerts for better spatial awareness.
  - Add user configuration for danger and caution thresholds directly from the app/dashboard.
- **Mobile Notifications:**
  Implement Blynk push notifications for critical events, such as dangerously close obstacles or low-battery alerts.
- **Hardware Robustness:**
  Develop integrated enclosures or custom PCBs for the ESP32 sensor nodes to improve durability, facilitate installation in vehicles, and reduce wiring complexity.
- **Data Logging and Export:**
  Track near-miss events or parking sessions for safety analysis and potential integration with driving history or accident prevention analytics.

8. Implementation & Testing

- **Assembly:**
  System hardware built on breadboards, validated for pinout, sensor mounting, and logical layout (see - wiring diagram).
- **Firmware:**
  Sensor and controller codes developed in Arduino IDE; thoroughly tested for reliable data flow, real-time feedback, and stable dashboard and buzzer performance.
- **Testing:**
  Functional system subjected to multiple obstacle scenarios, confirming zone response, warning logic, and successful data transmission across both dashboards.
- **Documentation:**
  All results, key screenshots (mobile/web), code, and photos preserved for submission

9. YouTube Demo Link

YouTube Channel –

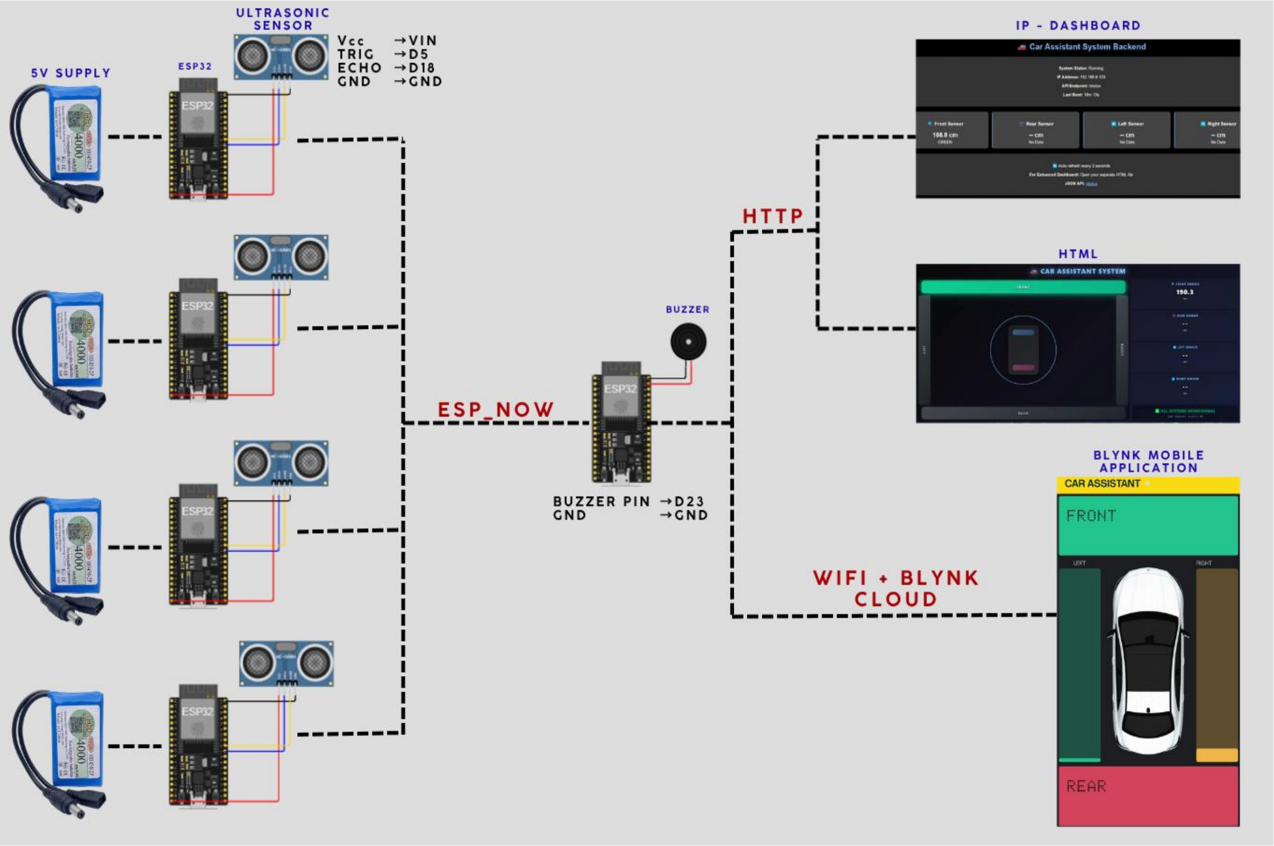
https://www.youtube.com/channel/UCuuQPkeXlVICukqogf8g7dQ

YouTube Thumbnail-

YouTube Video Link-

https://youtu.be/x9m71kpAn_o

10. Wiring Diagram



11. GitHub Link

https://github.com/Sumudika-Dilshan/car-parking-assistant-iot

12. References

1) Project Proposal
2) https://www.youtube.com
3) https://blynk.cloud/dashboard/836286/get-started
4) https://en.wikipedia.org/wiki/Ultrasonic_transducer
5) https://www.canva.com/design/DAGudrjraxk/jYUoYV_SHDuwyprhecXmUw/view?utm_content=DAGudrjraxk&utm_campaign=designshare&utm_medium=link2&utm_source=uniquelinks&utlId=hcfd47d5992