



Module Code & Module Title
CS5068NI– Cloud Computing & IoT
<<Smart Parking System>>

Assessment Type
10% Proposal Report / 50% Group Report
Semester
2025 Spring

Group Members

London Met ID	Student Name
23047596	Pratik Satyal
23047570	Aayush Raj Kafle
23047578	Kishor Baka Magar
23048526	Kriti Pun
23047597	Sobika Pradhan
23047594	Prekchhya Bimali

Assignment Due Date: 15 May,2025
Assignment Submission Date:15 May 2025
Submitted to: Mr. Sugat Man Shakya
Word Count: 4972

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

Turnitin 1

23047596 Pratik Satyal L2N8 Group1.docx

 Islington College, Nepal

Document Details

Submission ID

trn:oid:::3618:95954538

Submission Date

May 15, 2025, 7:58 AM GMT+5:45

Download Date

May 15, 2025, 7:59 AM GMT+5:45

File Name

23047596 Pratik Satyal L2N8 Group1.docx

File Size

31.2 KB

32 Pages





4,940 Words

26,085 Characters




7% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Match Groups

-  **27 Not Cited or Quoted 6%**
Matches with neither in-text citation nor quotation marks
-  **6 Missing Quotations 1%**
Matches that are still very similar to source material
-  **0 Missing Citation 0%**
Matches that have quotation marks, but no in-text citation
-  **0 Cited and Quoted 0%**
Matches with in-text citation present, but no quotation marks

Top Sources

- 3%  Internet sources
- 0%  Publications
- 6%  Submitted works (Student Papers)

Integrity Flags





0 Integrity Flags for Review

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.




A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.



Match Groups

-  **27 Not Cited or Quoted 6%**
Matches with neither in-text citation nor quotation marks
-  **6 Missing Quotations 1%**
Matches that are still very similar to source material
-  **0 Missing Citation 0%**
Matches that have quotation marks, but no in-text citation
-  **0 Cited and Quoted 0%**
Matches with in-text citation present, but no quotation marks

Top Sources

- 3%  Internet sources
- 0%  Publications
- 6%  Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Internet	
steergroup.com		<1%
2	Submitted works	
University of Wales Institute, Cardiff on 2025-03-09		<1%
3	Internet	
www.ece.ucf.edu		<1%
4	Submitted works	
Florida Gulf Coast University on 2024-04-08		<1%
5	Submitted works	
University of Malaya on 2022-06-21		<1%

Acknowledgement

We would like to take a moment to thank our module leader as well as lecturer Mr. Sugat Man Shakya for giving us this opportunity to work on this IOT project on the topic Smart Parking System. This project has been incredibly valuable for our learning journey, and we are grateful for the chance to be part of it.

We would also like to extend our appreciation towards our tutor Mr. Bishnu Pandey for all the help and guidance he provided during this project. His support and advice have been incredibly valuable to us as a student, and we are thankful for his assistance.

Moreover, we are grateful to Islington College and LMU for providing facilities and resources that have been crucial for our project and the successful completion of this project. At last, we would like to extend our heartfelt gratitude to everyone who have contributed on this project and offered their encouragement.

Abstract

This report presents the design and implementation of a prototype Smart Parking System utilizing Internet of Things (IoT) technologies to address common urban parking challenges. The system employs Arduino Uno microcontrollers, infrared (IR) sensors, servo motors, and LED displays to detect vehicle presence, monitor real-time parking slot availability, and automate barrier control. The prototype demonstrates how affordable and readily available components can be integrated to create a scalable, energy-efficient parking management solution suitable for environments such as hospitals, educational institutions, and public parking areas. By providing real-time visual feedback and automated guidance, the system aims to minimize the time drivers spend searching for available spaces, thereby reducing traffic congestion, fuel consumption, and user frustration. The report details the system architecture, hardware and software components, and testing results, highlighting the potential for future integration with mobile applications and advanced IoT-based features such as remote monitoring, automatic billing, and smart navigation. This approach offers a practical, low-cost alternative to existing complex and expensive parking solutions, with a focus on usability, modularity, and scalability for modern urban needs.

Table of Contents

1. Introduction	1
1.1 Scope and Usability of the project.....	1
1.2 Current Scenario	2
1.3 Aim and Objectives	3
1.4 Problem statement and its Solutions.....	4
1.4.1 Inefficient Space Utilization	4
1.4.2 Traffic Congestion and Fuel Waste.....	4
1.4.3 Expensive and Complex Existing Solutions	4
1.4.4 Driver Inconvenience and Frustration.....	4
1.4.5 Lack of Real-Time Information	4
2. Background	6
2.1 System overview	6
2.2 Design Diagrams	6
2.2.1 Block Diagram.....	7
2.2.2 System Architecture	8
2.2.3 Circuit Diagram	9
2.2.4 Schematic Diagram.....	10
2.2.5 Flowchart	11
2.3 Requirement Analysis.....	12
2.3.1 Hardware Components.....	12
2.3.2 Software components	16
3. Development	18
3.1 Input output system development and code	18
3.1.1 Step 1: Design and Planning.....	18
3.1.2 Step 2: Resource Collection.....	18
3.1.3 Step 3: System Development	19
3.2 Connections of Pins to Arduino and Breadboard.....	23
4. Results and Findings	25
Test 1: Check for codes operability	25
Test 2: Check for exit and entry operability	26
Test 3: Check for Slot 1 and Slot 2 status.....	31

Test 4: Check for the try again message when parking is full.	33
Test 5: Check if barrier opens when the slots are full but the occupancy free status.	34
Test 6: Check if occupancy goes below the available number of slot count	35
Test 7: Check if occupancy goes above the available number of slot count.	36
5. Future Works.....	39
5.1 Sustainable Smart Parking System.....	39
5.2 Electric Vehicle (EV) Support	39
5.3 Enhanced Security & Surveillance	40
5.4 Integration with Autonomous Vehicles	40
6. Conclusion.....	42
7. References	43
8. Appendix	46
8.1 Individual contribution Plan.....	46
8.2 Code used for Smart Parking System.....	47

Table of Figures

Figure 1: Outdated parking system (TheAseanPost , 2019)	2
Figure 2: INRIX Parking Ranking – Hours Spent Searching for Parking (INRIX Research, 2017).	3
Figure 3:Managed Parking System (Smart Life, 2022)	5
Figure 4: Block Diagram	7
Figure 5: System Architecture	8
Figure 6: Circuit Diagram	9
Figure 7: Schematic Diagram	10
Figure 8: Flowchart	11
Figure 9: Arduino Uno (Teja, 2024)	12
Figure 10: Breadboard (Spiceman, 2022)	13
Figure 11: Servo Motor (Apoorve, 2025)	14
Figure 12: Jumper wire (element14, 2025)	15
Figure 13: IR sensor (Watson, 2023)	16
Figure 14: Arduino IDE (Anderson, 2012).....	16
Figure 15: Tinkercad (AutoDesk, 2015)	17
Figure 16: Draw.io (link, 2025)	17
Figure 17: Phase 1- Connecting input devices (IR SENSORS) to Arduino	19
Figure 18: Phase 2- Connecting Output Devices (I2C and Servo Motor) to Arduino	20
Figure 19: Phase 3- Connecting power supplies (Bread Board and 4x1.5 Volt Batteries) to each component	22
Figure 20 : Pushing the Code inside Arduino Board for Functionality	22
Figure 21: Final Connected Image	24
Figure 22: Selecting correct port and word	25

Figure 23: Successfully uploaded the code.....	26
Figure 24: Barrier opening at entry.....	27
Figure 25: Barrier closes at entry	28
Figure 26: Barrier opening at exit	29
Figure 27: Barrier closes at exit.....	30
Figure 28: Checking when S1 is occupied	31
Figure 29: Checking when S2 is occupied	32
Figure 30: Checking when both S1 and S2 is occupied.....	32
Figure 31: Parking full message when at maximum occupancy	33
Figure 32: Barrier didn't open.....	34
Figure 33: Slot counter at its maximum occupancy before	35
Figure 34: Slot counter at its maximum occupancy after	36
Figure 35: Slot Counter at its minimum occupancy before.....	37
Figure 36: Slot Counter at minimum occupancy after	38

Table of Tables

Table 1: Arduino Uno Specifications	12
Table 2: Breadboard Specifications	13
Table 3: Servo motor Specifications.....	14
Table 4: Jumper Wires Specifications.....	15
Table 5: IR sensor Specifications	16
Table 6: Pin Connection.....	23
Table 7: Checking for code operability.....	25
Table 8: Check for exit and entry operability	26
Table 9: Check for Slot 1 and Slot 2 status	31
Table 10: Check for the try again message when parking is full.....	33
Table 11: Check if barrier opens when the slots are full but the occupancy free status	34
Table 12: Check for the minimum number of Slots.	35
Table 13: Check for the minimum number of Slots	36

Smart Parking System

1. Introduction

The Internet of Things (IoT) is a network of devices that collect and share data using the help of sensors and software (Perwej, et al., 2019). Using IoT in this project, real-time monitoring and automation allow the Smart Parking System to detect vehicles, monitor parking slots and control barriers automatically.

With the rapid modernization of the transportation industry, every house has a transportation medium, the normalization of owning a transportation medium i.e. cars and bikes have generated a new problem with managing the parking for those mediums. Especially in the urban/city areas this has been an uprising issue, this Smart Parking System aims to aid that very parking problem, with sensors and automated doors along with slot counters, this project has the potential to completely replace the traditional parking lot management where human resources are required.

In the project, the infrared sensors are designed and implemented to detect the vehicle presence and to give visual feedback on the availability of slots. The system automates the parking process and can be useful for hospitals, educational institutions and public parking places. This prototype is simulated in the sense that it utilizes technologies like LEDs, Infrared sensors, motors, while utilizing them to demonstrate integral smart parking functionality that can be scaled.

1.1 Scope and Usability of the project

The smart parking system scope includes intelligent parking prototype development, real time parking space availability demonstration and its simulation. And after that, through implemented components like Arduino Uno, IR sensors, the mechanism is created.

While designing the system, future scalability is in mind. It can be integrated with mobile applications, IoT based monitoring and remote management system and now the users can book and track the parking slots in real time. The project provides a structure for future features like automatic billing, security, monitoring and smart navigations. The system is designed using affordable components, is flexible and modular. The system is designed by using the fewest resources possible without reducing functionality.

The system automatically detects vehicle presence and shows all the real time spots availability which helps drivers to avoid the hassle of searching for available parking space, consuming less fuel and reducing traffic congestion. This system is especially useful in environments like shopping malls,

hospitals, office buildings, and residential buildings where efficient space management is crucial which makes it a highly useful solution.

1.2 Current Scenario

Currently in many developed or underdeveloped countries, parking is a common problem which they are facing. The process of parking is either fully managed or manual with attendants, systems are mostly paper based which slows the process down and brings human error and inefficiency. In this digital era, some parking systems are based on digital apps, but not all parking apps always reflect real time data which might delay or confuse an individual. This is also very much a problem in high density cities, affecting daily commutes and business productivity.



Figure 1: Outdated parking system (*TheAseanPost*, 2019).

According to research by INRIX, U.S. drivers alone spend an average of 17 hours per year looking for parking, costing approximately \$345 per driver annually in wasted time, fuel, and emissions (INRIX Research, 2017).

Rank	U.S. City	Average 2-Hour Parking Cost (One mile of city center)	On-Street Search Time (mins/trip)	Off-Street Search Time (mins/trip)	Annual Search Time (hours/driver/year)	Annual Search Cost Per Driver	Annual Search Cost Per City
1	New York	\$33	15	13	107	\$2,243	\$4.3bn
2	Los Angeles	\$14	12	11	85	\$1,785	\$3.7bn
3	San Francisco	\$12	12	11	83	\$1,735	\$655m
4	Washington D.C.	\$18	10	9	65	\$1,367	\$329m
5	Seattle	\$10	9	8	58	\$1,205	\$490m
6	Chicago	\$22	9	8	56	\$1,174	\$1.3bn
7	Boston	\$26	8	8	53	\$1,111	\$262m
8	Atlanta	\$6	8	8	50	\$1,043	\$251m
9	Dallas	\$6	8	8	48	\$995	\$726m
10	Detroit	\$9	6	6	35	\$731	\$209m
	US	\$4	2	2	17	\$345	\$72.7bn

Figure 2: INRIX Parking Ranking – Hours Spent Searching for Parking (INRIX Research, 2017).

1.3 Aim and Objectives

Aim:

To develop a prototype of Smart Car Parking System for some real-time slot detection, automated guidance and display through embedded systems and sensor technologies.

Objectives:

- To design and build a functionally based model that uses IR sensors and Arduino uno to detect the presence of the vehicles.
- To provide real-time visual indicators for slot availability.
- To increase user convenience by reducing the user's time spent searching for parking spaces.
- To develop the system architecture that can scale, build, or integrate with mobile or IoT platforms in future development phases.
- To test the system through real time testing and to validate its effectiveness under conditions simulated.
- To ensure low-cost implementation using readily available and energy-efficient components.

1.4 Problem statement and its Solutions

The increasing number of vehicles is the main cause of demand for the efficient parking system which is also driven by urbanization. Traditional parking systems are not very suitable, and they don't fulfill the needs of modern days. Some problems which were identified:

1.4.1 Inefficient Space Utilization

Problem: Due to the lack of management, many parking areas are poorly organized, which results in blocked or unused space.

Solution: The smart parking system uses IR sensors to detect each slot, ensuring that only available spaces are shown.

1.4.2 Traffic Congestion and Fuel Waste

Problem: Searching for parking increases road traffic, fuel consumption, and carbon emissions.

Solution: The smart parking system reduces the time spent finding a spot, helps minimize vehicle movement, save fuel usage and minimize pressure.

1.4.3 Expensive and Complex Existing Solutions

Problem: Most smart systems available today are high in cost and require large-scale infrastructure changes.

Solution: The smart parking system offers a low-cost, small-scale prototype using readily available components, ideal for areas that need affordable and scalable solutions.

1.4.4 Driver Inconvenience and Frustration

Problem: Parking is uncertain, what causes stress, delays, and user dissatisfaction.

Solution: The smart parking system makes the process faster and more convenient by providing clear visual instructions which improves user experience.

1.4.5 Lack of Real-Time Information

Problem: Before entering, drivers don't know which parking space is available, which may lead to unnecessary delays.

Solution: The smart parking system detects real-time slot availability and displays it through digital indicators which will guide drivers directly to the available space if one is present.

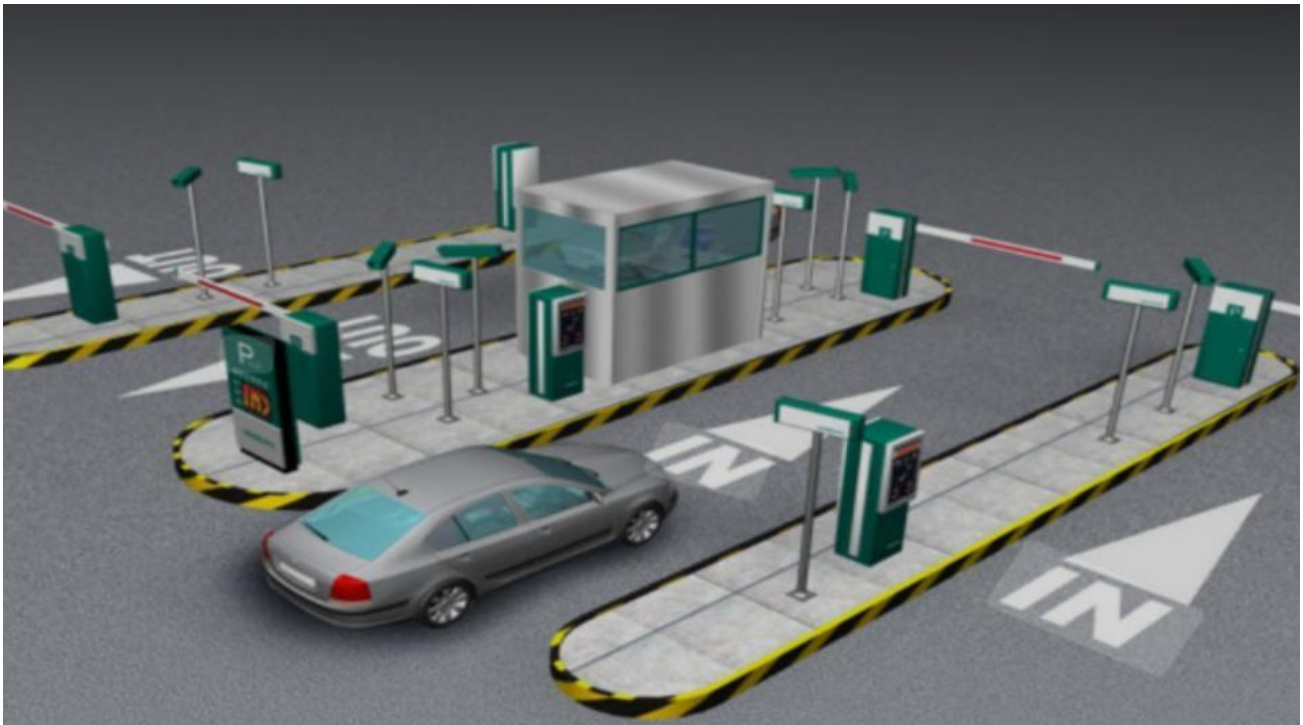


Figure 3:Managed Parking System (*Smart Life, 2022*).

2. Background

This section shows various main components and the structure upon which the Smart Parking System is based. It gives an overview of how these components such as Arduino, infrared sensors, servo motors, and displays work together in order to allow an automated parking system. The subsections demonstrate the system's layout, design diagrams, component functions, and software tools used to simulate and implement the project.

2.1 System overview

The project contains an Arduino Board, Servo Motor and IR sensors connected through breadboard and jumper wires to reach the required goal of this project. The IR works as an input that is an active component that detects vehicles and sends signals to the Arduino Uno then the Arduino sends the signal to the LCD Display and the Servo which are the actuator according to the location of the Sensors and the description given to them. Jumper Wires are used to connect devices together and the breadboard is used as a bridge between devices to provide power supply to each devices except the servo through the Arduino Uno.

2.2 Design Diagrams

Here an example of diagrams that outline the hardware components, architecture, interconnections, and data movement of the system are provided.

2.2.1 Block Diagram

The block diagram shows the basic layout of the components that are required for the making of this IoT project “Smart Parking System”.

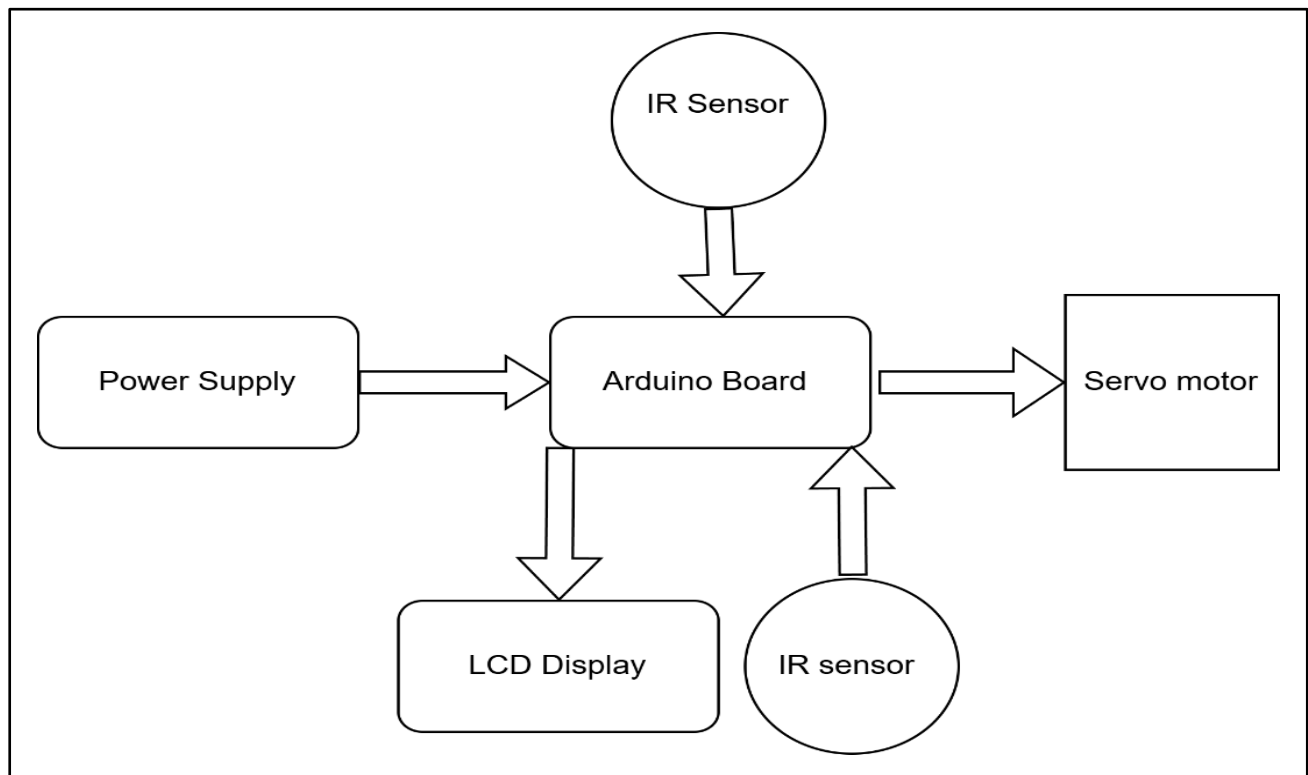


Figure 4: Block Diagram

At the center of the circuit is the Arduino Uno that connects all the actuators and sensors and ensures that the system operates as intended. The Arduino is powered by battery and through the microcontroller board both the IR Sensors and LED display are powered on brought the power supply source connected to the Arduino Uno.

2.2.2 System Architecture

The shown system Architecture shows how the Smart parking System basic layout and how each sensor and actuator will function.

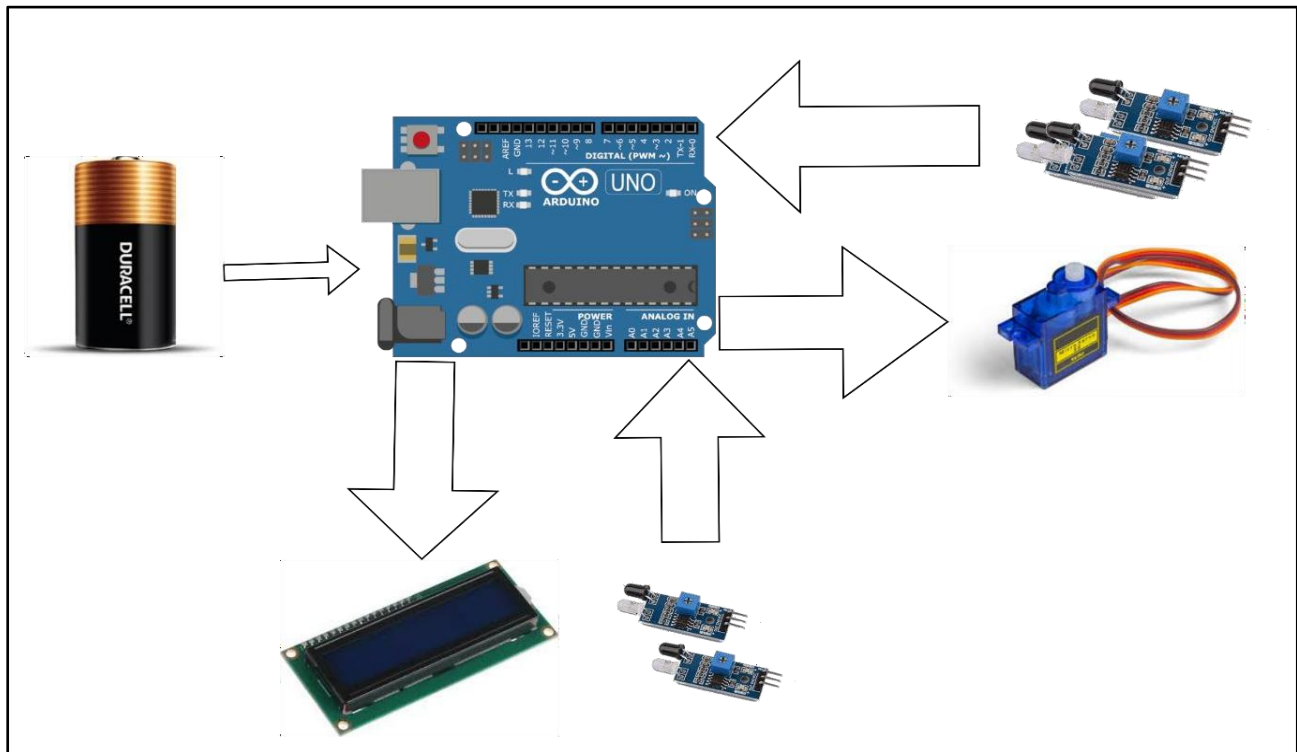


Figure 5: System Architecture

The system has 4 IR sensors with one Servo motor and one LED Display as actuators. At the start the 2 IR will open and close the barrier connected to the servo motor if the IR detect any vehicles and when the vehicle is parked in the parking spot it will update the available spots in the LED display accordingly and if the vehicle is removed from the parking spot it will again update the available number of parking spaces and when the vehicle is required to be removed from the parking lot it will again open the barrier using the 2 IR sensor at the front gate accordingly.

2.2.3 Circuit Diagram

The Arduino Uno's 5v pin is connected to positive rail of the breadboard and GND pin is connected to negative rail of the breadboard using male to male jumper wire to power the IR sensors and the LED display. The IR's and LED display's VCC pin is connected to Positive rails of the breadboard while the GND pin is connected to negative rails of the breadboard using female to male jumper wires. The servo motor power is connected a secondary 6v case that only provides power to the only Servo motor as the it consumes 4.8 - 6 V at 5 – 6 mA when idle and other components of the system is powered through a 9v battery directly connected to the bread board matching both the positive and negative ends of both battery and the breadboard. The LED Display's SDA and SCL pins are connected to A4 and A5 Analog pin respectively of the Arduino while the IR sensor at the main entry OUT pin is connected to 2 and 3 digital and parking spots IR's OUT pin are connected to 5 and 6 pins in the Arduino Digitals pins and servos data pin is connected to digital 4 in the Arduino Uno.

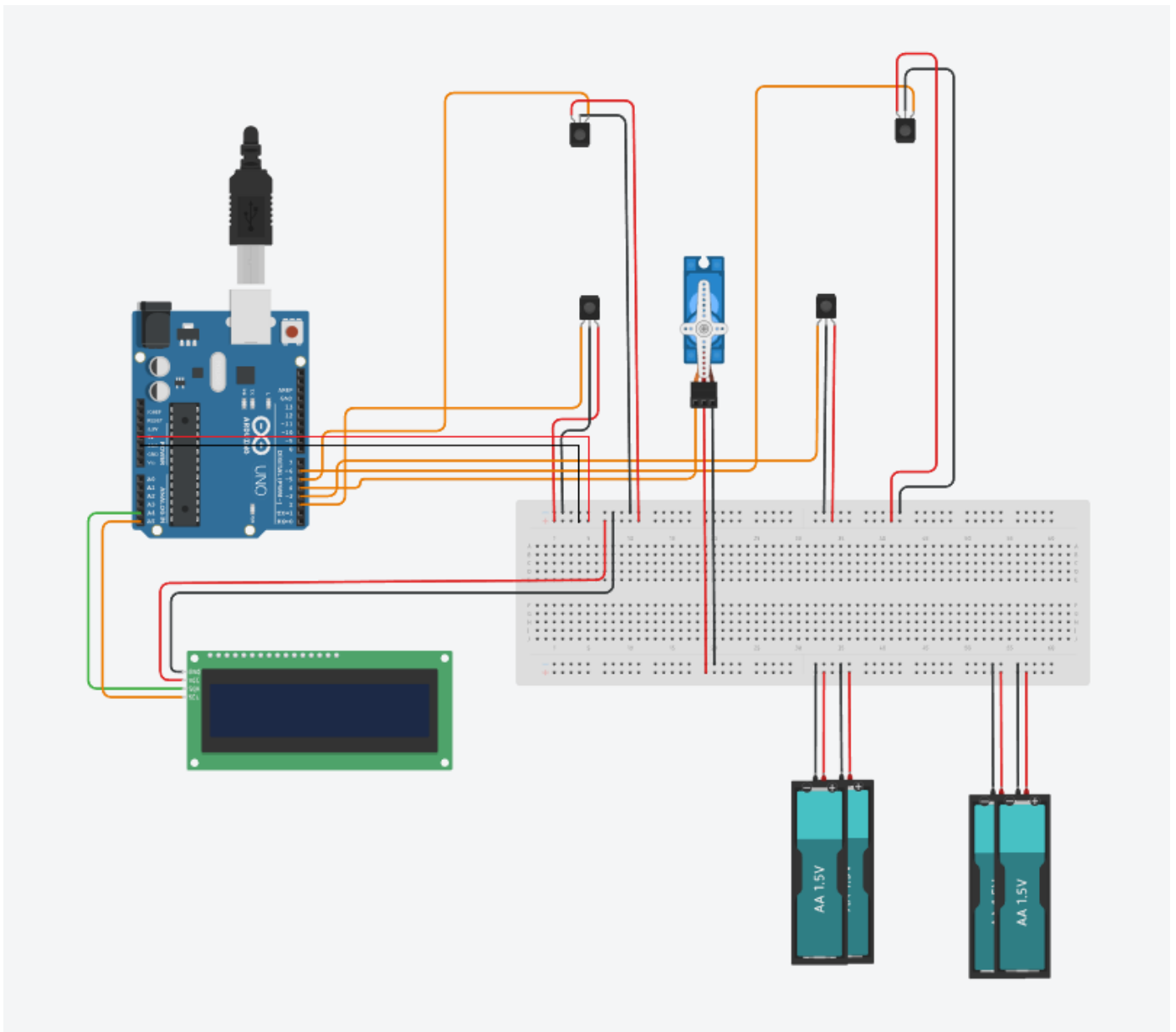


Figure 6: Circuit Diagram

2.2.4 Schematic Diagram

The diagram illustrates an Arduino Uno (Rev3) connected to four IR sensors (IR1-IR4) and a servo motor, with only the servo motor explicitly powered by a 6V source in this zoomed-in view.

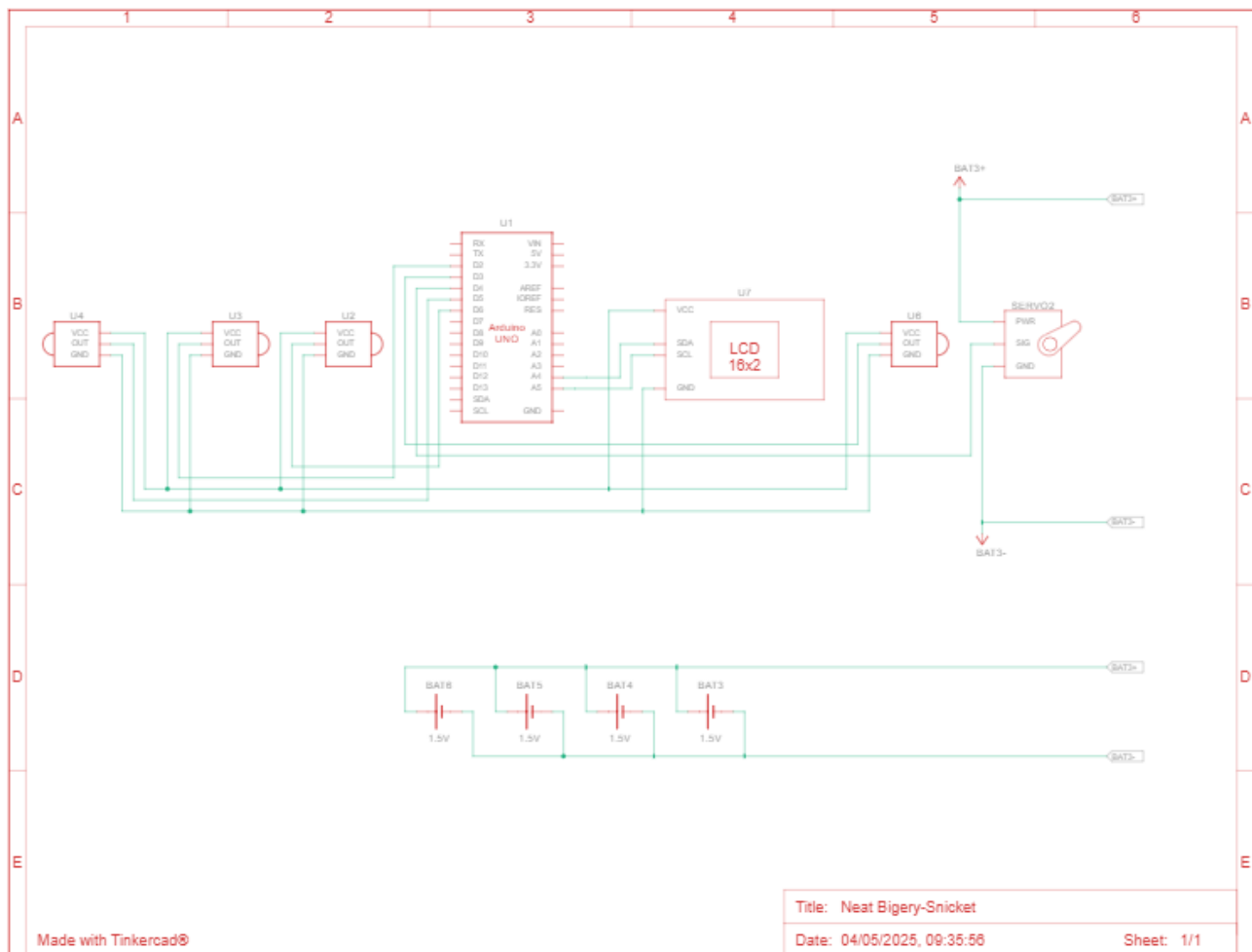


Figure 7: Schematic Diagram

The Arduino is configured to trigger each IR sensor, which then measures the time it takes for the reflected infrared light to return, thereby determining if an object is in close proximity. This proximity data, gathered from the multiple IR sensors, would be processed by the Arduino to generate control signals for the servo motor. In a parking system application, these IR sensors could be strategically placed to detect the arrival or departure of vehicles at entry/exit points or to monitor the occupancy of individual parking spaces. Based on the input from these sensors, the Arduino would then actuate the servo motor to control a barrier gate, allowing or denying access to the parking area.

2.2.5 Flowchart

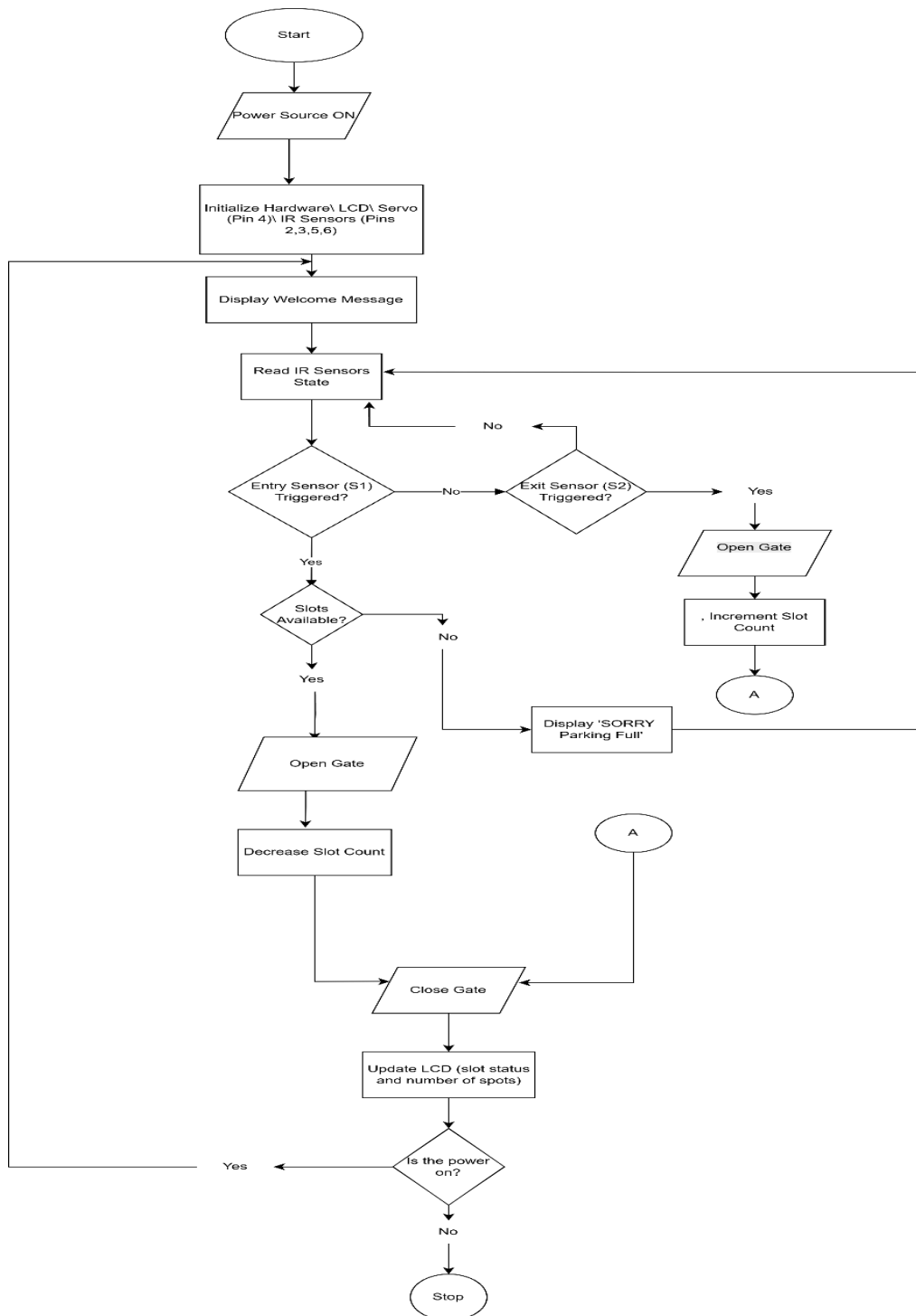


Figure 8: Flowchart

2.3 Requirement Analysis

2.3.1 Hardware Components

In this project, hardware components are mainly responsible for processing the signals, detection process, and performing physical actions like opening and closing a gate. The prime hardware that are used are like:

- **Arduino Uno**

An active digital processing component which is based on ATmega328P processor and can be implemented into many electronic projects. Arduino Uno is low cost, flexible open-source microcontroller board (Kumar, et al., 2015).

Function	In this system, it acts as the central processor which processes signals from input device sends appropriate command for output actions.
Type	Active
Signal Type	Digital

Table 1: Arduino Uno Specifications

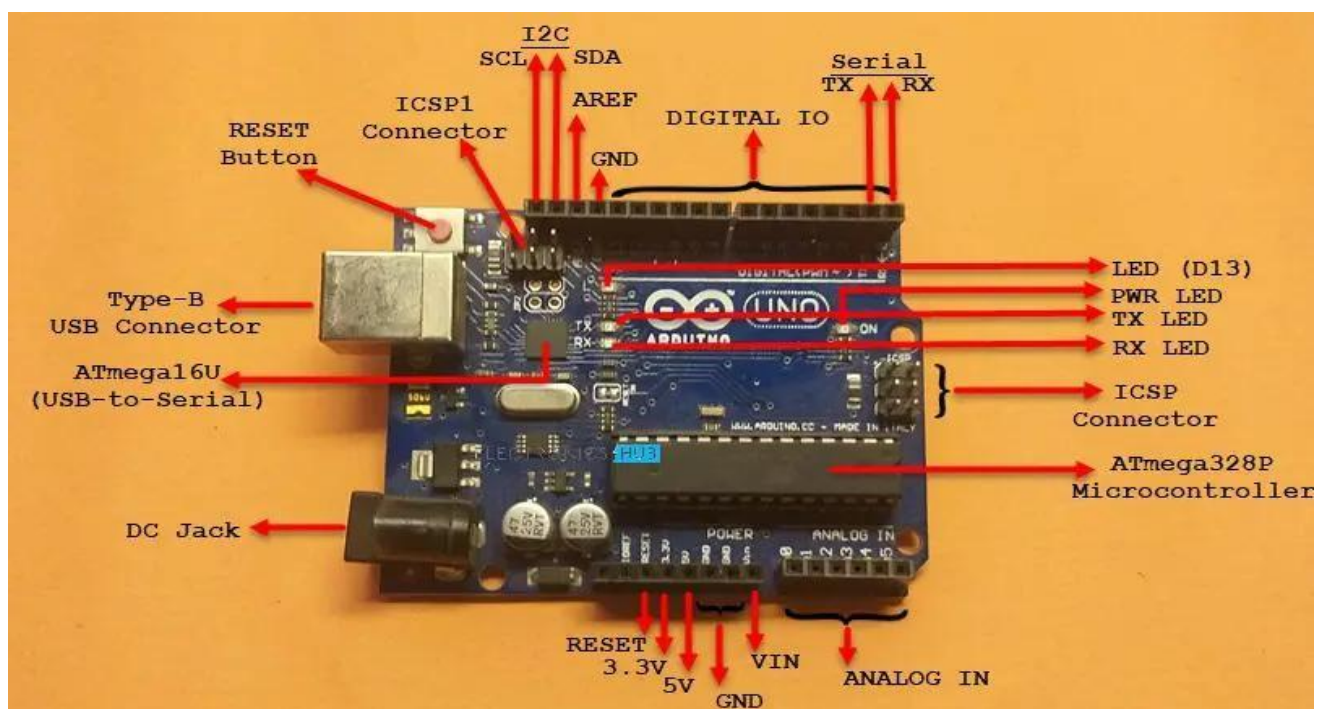


Figure 9: Arduino Uno (Teja, 2024).

▪ Breadboard

A passive prototyping tool which is the foundation to construct and prototype electronics. It allows temporary placements of components and connections on the board to make circuits without joining. It is non-electronic passive tool, so it doesn't process signals. (Crowell, 2019)

Function	In this system, it acts as a physical platform for connecting hardware components. (Physical Connector)
Type	Passive
Signal Role	None (non-electronic)

Table 2: Breadboard Specifications

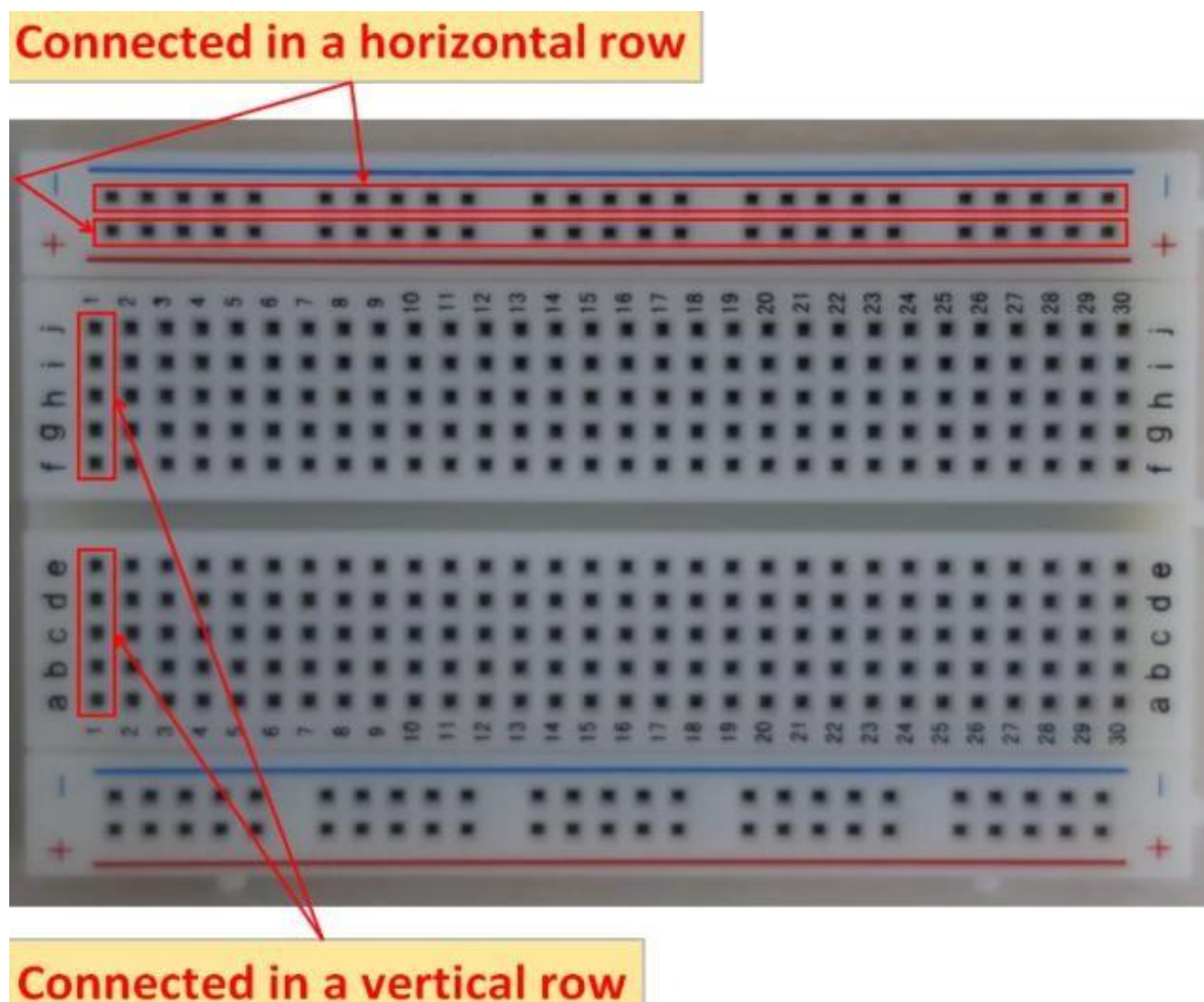


Figure 10: Breadboard (Spiceman, 2022).

▪ Servo motor

An active component that is a very specialized motor in control of rotary or linear motion (Firoozian, 2014).

Function	In this system, it acts as an actuator to open or close a gate based on digital signals (PWM) received from the microcontroller. {Output (Actuator)}
Type	Active
Signal Type	Digital

Table 3: Servo motor Specifications



Figure 11: Servo Motor (Apoorve, 2025).

▪ Jumper wires

A passive connector component which has connector pins at each end used for connecting two points (Lall & Pecht, 2018).

Function	In this system, it is used to make connections between components on the breadboard and Arduino. (Electrical Interconnection)
Type	Passive

Signal Role	Analog/Digital Carrier (Depends on usage)
-------------	---

Table 4: Jumper Wires Specifications

Figure 12: Jumper wire (*element14*, 2025).

▪ IR sensors

An active input component that emits the infrared light to sense some object of the surroundings (Rogalski & Chrzanowski, 2017) .

Function	In this system, it is used to detect the presence or absence of a vehicle in a specific parking slot. It also sends a digital (HIGH or LOW) signal to the microcontroller based on whether an object is detected. {Input (Object Detection)}
Type	Active

Signal Type	Digital
-------------	---------

Table 5: IR sensor Specifications

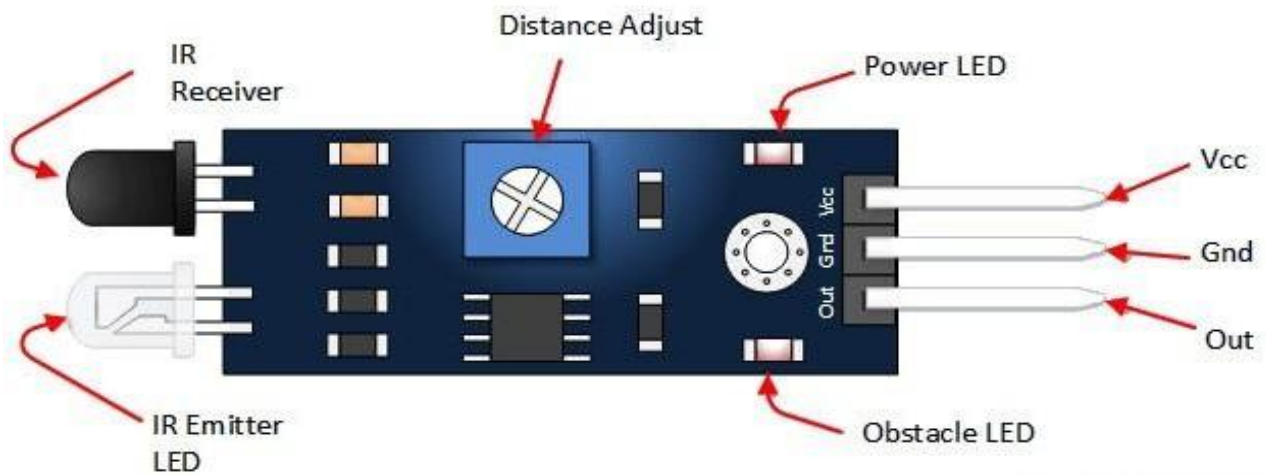


Figure 13: IR sensor (Watson, 2023).

2.3.2 Software components

▪ Arduino IDE

It is a platform that helps you to write, compile and upload the code to Arduino boards. It serves for the development environment for an embedded C/C++ code in what allows communication between software and hardware (ThinkRobotics, 2025). In this system, it uses to assign the Arduino Uno how to interact with the senses, motors.



Figure 14: Arduino IDE (Anderson, 2012).

- **Tinkercad**

Tinkercad was created by Autodesk which is a free and online 3D design and electronics simulator. This is widely used for learning, prototyping and teaching STEM concepts, particularly for beginners (Barbosa, et al., 2024). In this system, this is used for making diagrams like Schematic Diagram, Circuit Diagram.



Figure 15: Tinkercad (AutoDesk, 2015).

- **Draw.io**

It is used for creating system architecture diagrams, flowcharts, design layouts, etc. in an online diagramming tool (Kaplan & Rabelo, 2024). In this system, this is used to make diagrams like System Architecture, Block Diagram, Flowchart etc.



Figure 16: Draw.io (link, 2025).

3. Development

3.1 Input output system development and code

The development section includes process of development of the IOT project.

3.1.1 Step 1: Design and Planning

Initially, a group discussion was held which finalized the project of Smart Parking System, the system was simulated on tinker cad and a conclusion was met that the project would be made with Arduino Uno acting as the central controller, while Infrared Sensors will aid and act as the detecting units for the parking system's parking slot and entrance barrier, A server moto was used to automate the entrance of the parking system and an LCD display was used to display the information about parking slots in the parking system.

3.1.2 Step 2: Resource Collection

After the design was finalized, a letter listing all the required components was presented to the module tutor, this letter aided in gathering the components from the college resource office, the components included:

- Arduino UNO
- Infrared (IR) Sensors
- Servo Motor
- I2C Display
- Breadboard

But due to the unavailability of sufficient jumper wires on college premises they were collected from a nearby store.

3.1.3 Step 3: System Development

The main function of the parking system was to detect the among of vehicles inside the parking system and notify the users if there was any slot open, while automating any required tasks like barrier entrances. The development of this project has been divided into 5 phases.

Phase 1: Connecting input devices (IR SENSORS) to Arduino

In this phase connection from Arduino to 4 individual IR Sensors was made, The IR sensor's Out Pin's were connected To Arduino D2/3/5/6 Female Pins.

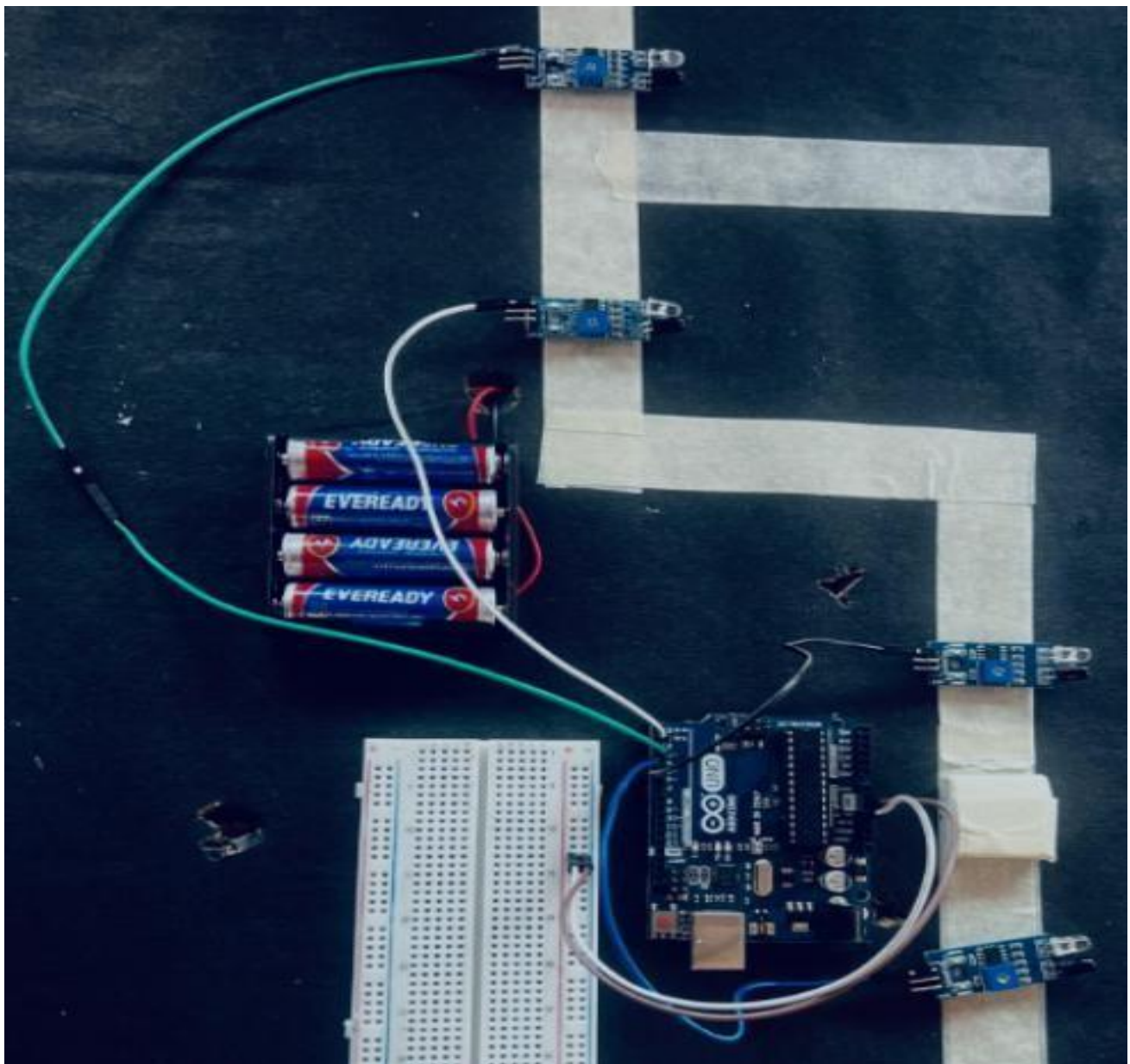


Figure 17: Phase 1- Connecting input devices (IR SENSORS) to Arduino

Phase 2: Connecting Output Devices (I2C and Servo Motor) to Arduino

In This Phase connection from Arduino to Servo Motor in Female Pin D4 and Connection from Arduino to I2C SCL and SDA port in A5 and A4 Arduino points was made respectively.

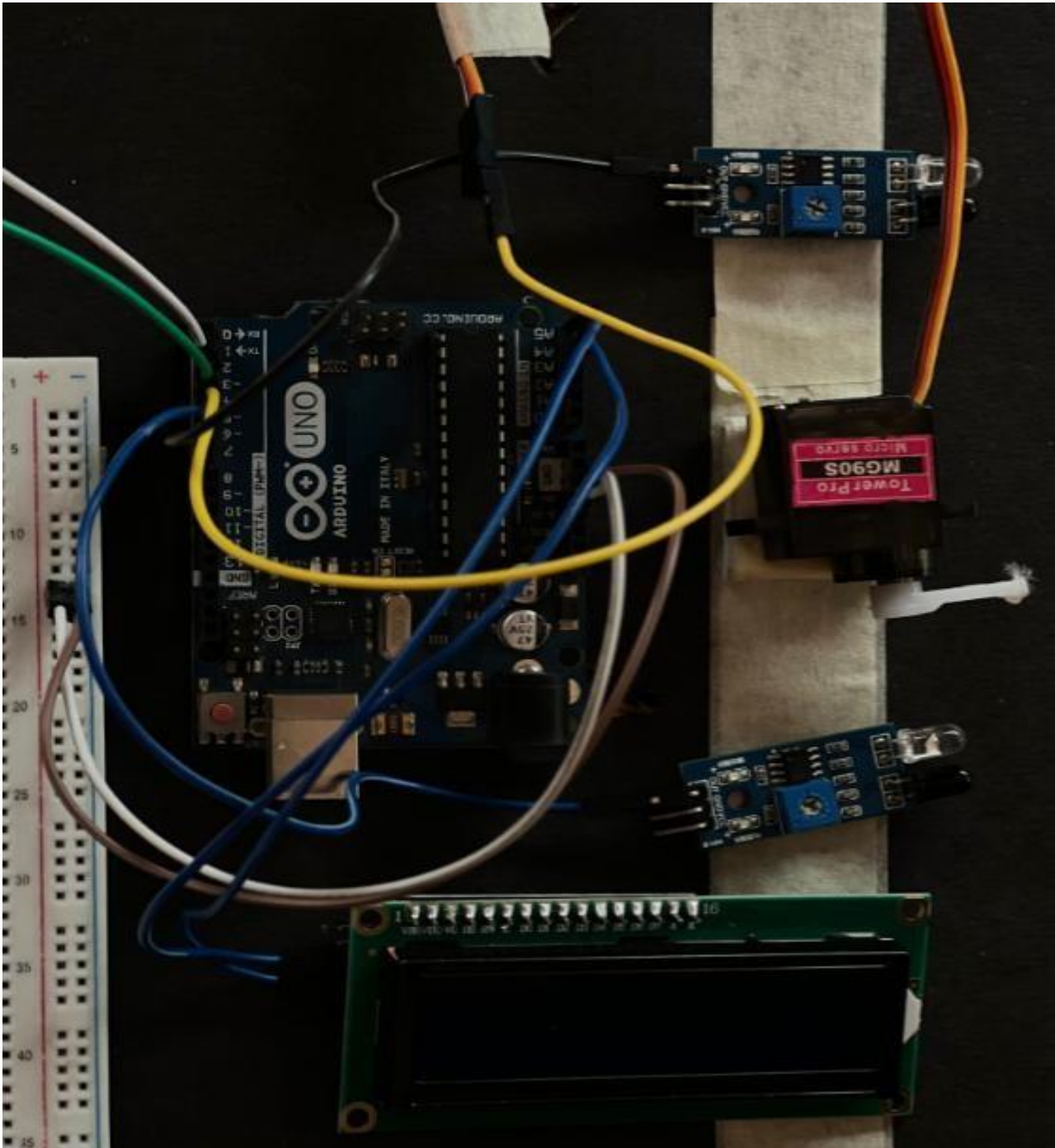
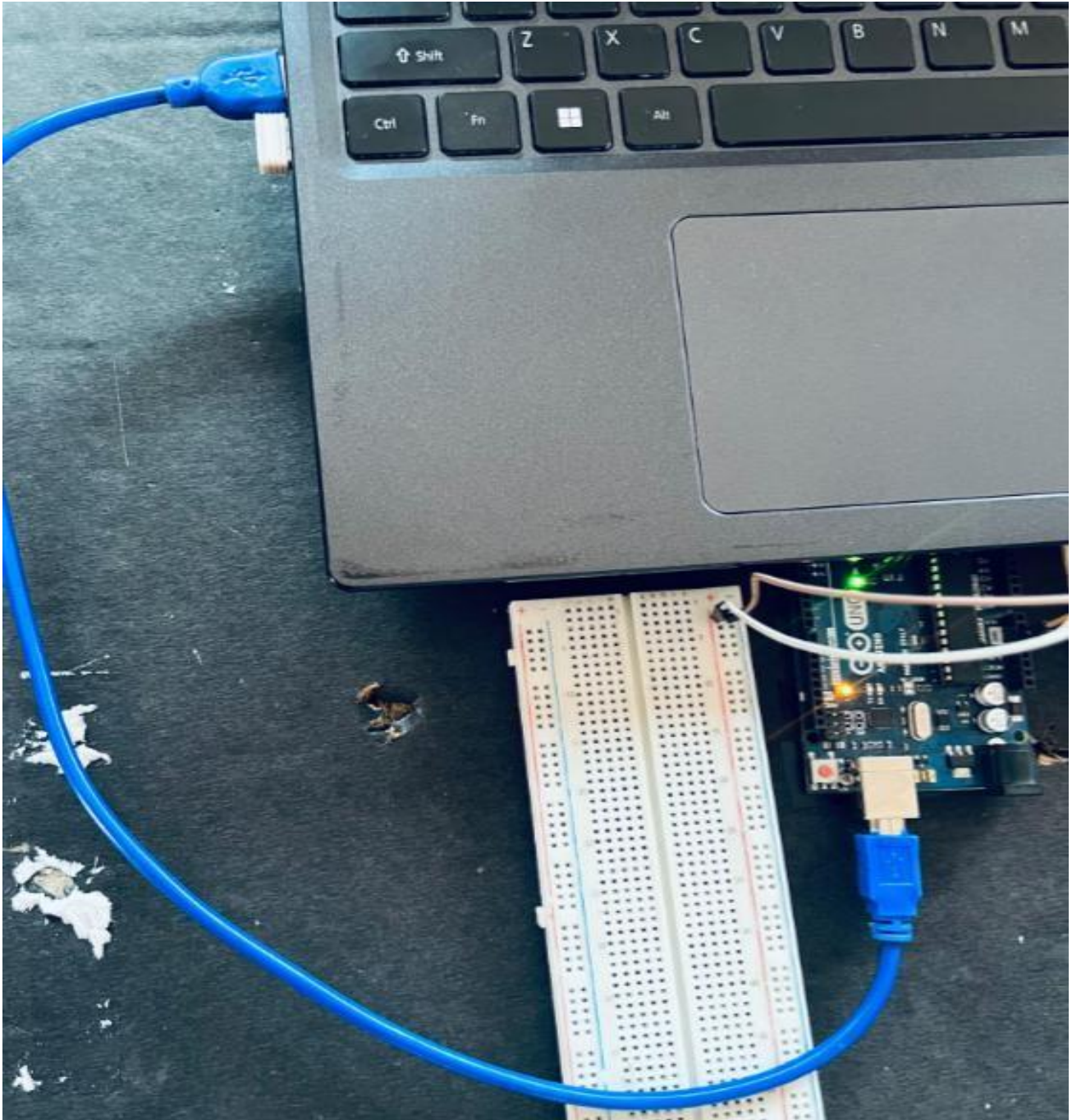


Figure 18: Phase 2- Connecting Output Devices (I2C and Servo Motor) to Arduino

Phase 3: Connecting power supplies (Bread Board and 4x1.5 Volt Batteries) to each component

In this phase first a connection to Arduino by laptop, the connection was made using console cable, and then a Arduino 5V and GND port was connected to breadboard positive and negative rail respectively, and after the connection was made each component except servo motor's VCC pin was connected to bread board positive rail and GND pin into negative rail, servo motor was connected to a 6Volt power output externally.



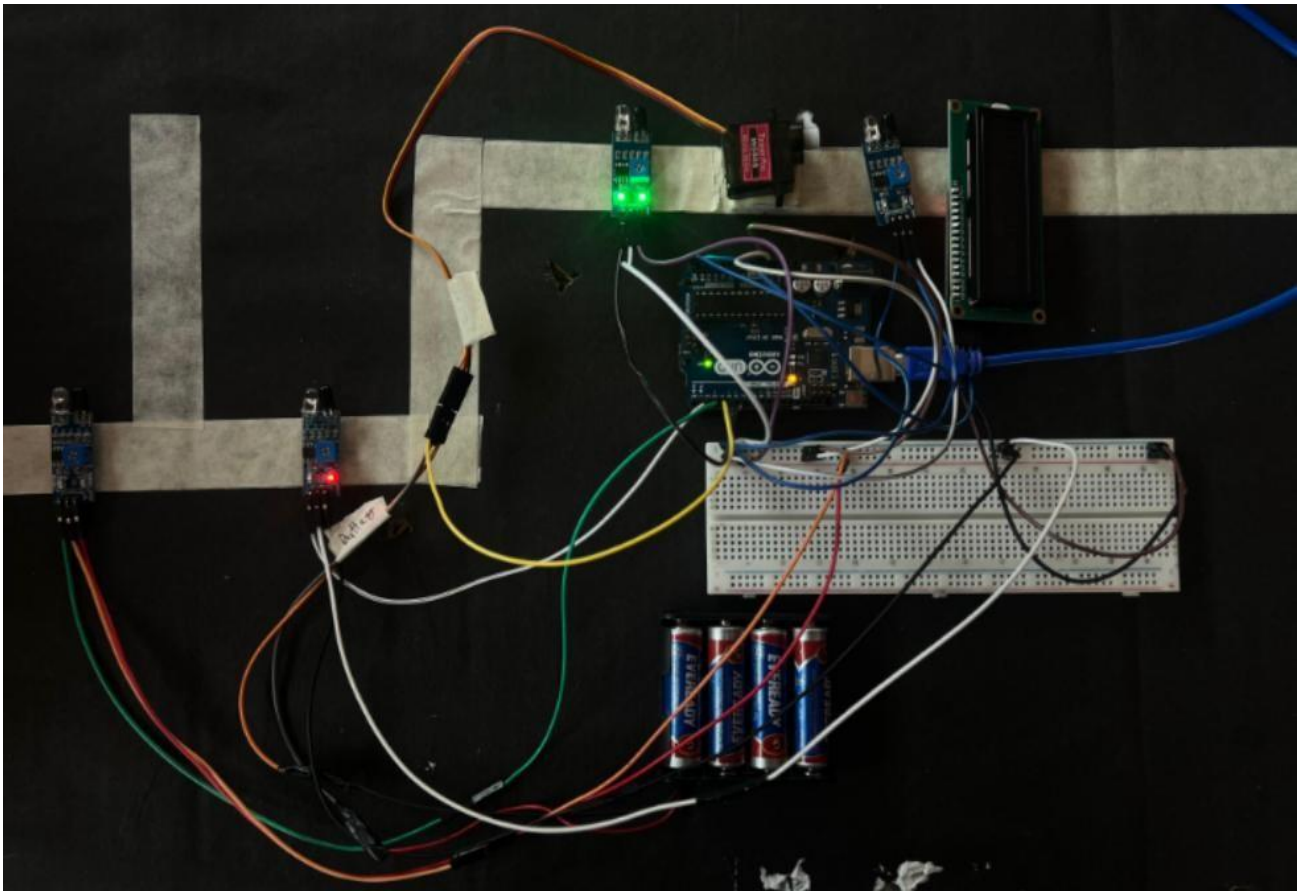


Figure 19: Phase 3- Connecting power supplies (Bread Board and 4x1.5 Volt Batteries) to each component

Phase 4: Pushing the Code inside Arduino Board for Functionality

Finally, the code was pushed into Arduino Board for the functionality of the project.

```
iotproject [Arduino IDE 2.1.6]
File Edit Sketch Tools Help
Arduino Uno
iotproject.ino
1 #include <Wire.h>
2 #include <LiquidCrystal_I2C.h>
3 #include <Servo.h>
4
5 // LCD setup
6 LiquidCrystal_I2C lcd(0x27, 16, 2);
7
8 // Servo setup
9 Servo myservo;
10
11 // IR sensor pins
12 const int IR1 = 2;
13 const int IR2 = 3;
14 const int IR3 = 5;
15 const int IR4 = 6;
16
17 // Parking slots
18 int Slot = 4; // Total number of parking slots
19
20 // Flags for sensors
21 int flag1 = 0;
22 int flag2 = 0;
23
24 void setup() {
25   Serial.begin(9600);
26
27   lcd.init(); // Initialize LCD
28   lcd.backlight(); // Turn on backlight
29
30   // Setup IR sensor pins
31   pinMode(IR1, INPUT);
32   pinMode(IR2, INPUT);
33   pinMode(IR3, INPUT);
34   pinMode(IR4, INPUT);
35
36   // Setup servo
37   myservo.attach(4); // Servo connected to pin 4
38   myservo.write(180); // Initial position (closed)
39
40   // Welcome message
41   lcd.setCursor(0, 0);
42   lcd.print(" ARDUINO ");
43   lcd.setCursor(0, 1);
44   lcd.print(" PARKING SYSTEM ");
45   delay(2000);
46   lcd.clear();
47 }
```

Figure 20 : Pushing the Code inside Arduino Board for Functionality

3.2 Connections of Pins to Arduino and Breadboard

Arduino is powered by USB cable and is connected to a laptop. Servo Motor is powered by external 6V battery.

▪ Pin Connection

Device	Pins	Connected Device	Pins
Arduino	5V	Breadboard	Positive
	GND		Negative
IR Sensor 1-4	VCC	Breadboard	Positive
	GND		Negative
	OUT	Arduino Uno	D2/D3/D5/D6
Servo Motor	Positive	Battery	Positive
	Negative		Negative
	Data	Arduino Uno	D4
LCD Display	VCC	Breadboard	Positive
	GND		Negative
	SCL	Arduino Uno	A5
	SDA		A4

Table 6: Pin Connection

- **Final Connected Image**

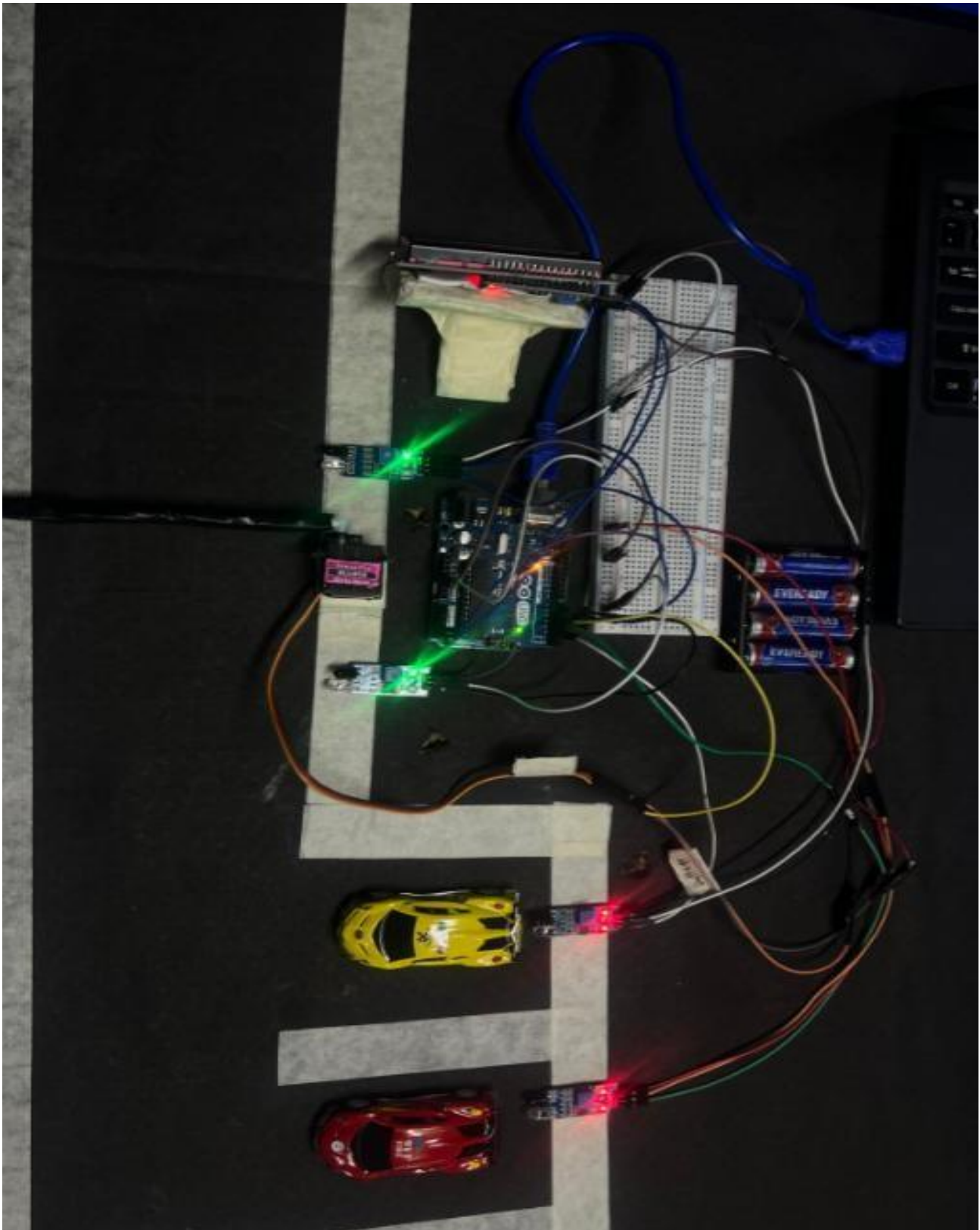


Figure 21: Final Connected Image

4. Results and Findings

This part provides outcomes of various tests carried out to evaluate the functionality, accuracy, and reliability of the Smart Parking System prototype. The role of every test was to ensure that specific features such as functionality of the barrier, accuracy in identification of slots, as well as compliance of the system within specified limits, had been confirmed. The results give us an idea of how the system behaves in conditions that create an illusion of the real parking environments.

Test 1: Check for codes operability

Test No	1
Objective	To check if the code is operatable and able to be uploaded into the Arduino uno without any problems.
Activity	Select related port and Board from the tools tab the Compile and Upload the code to the Arduino UNO
Expected result	The code should be easily compiled and uploaded.
Actual Result	The code was compiled and uploaded.
Conclusion	The test was Successful.

Table 7: Checking for code operability

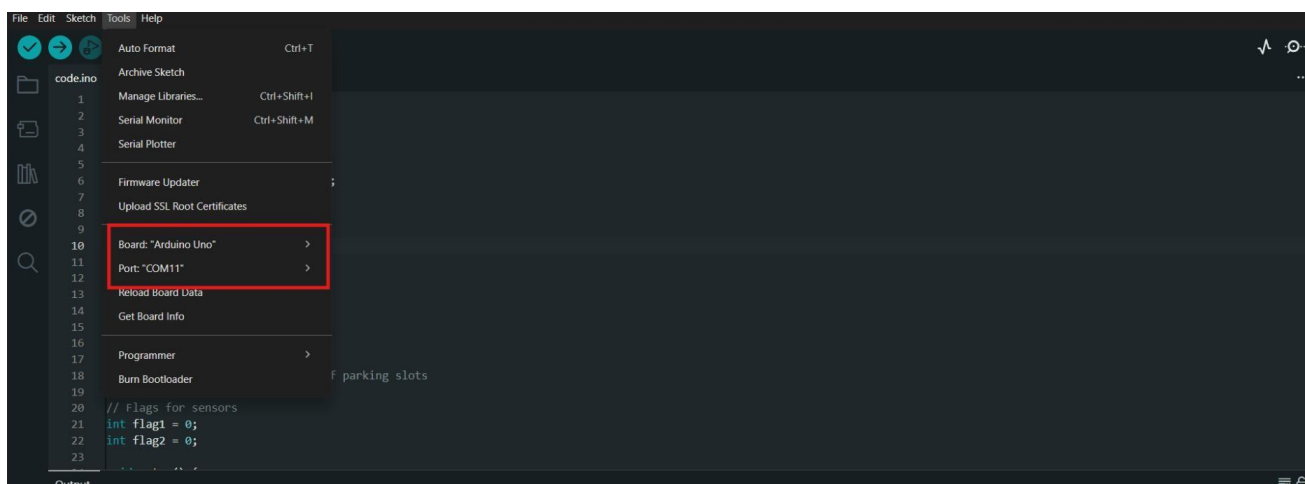


Figure 22: Selecting correct port and word

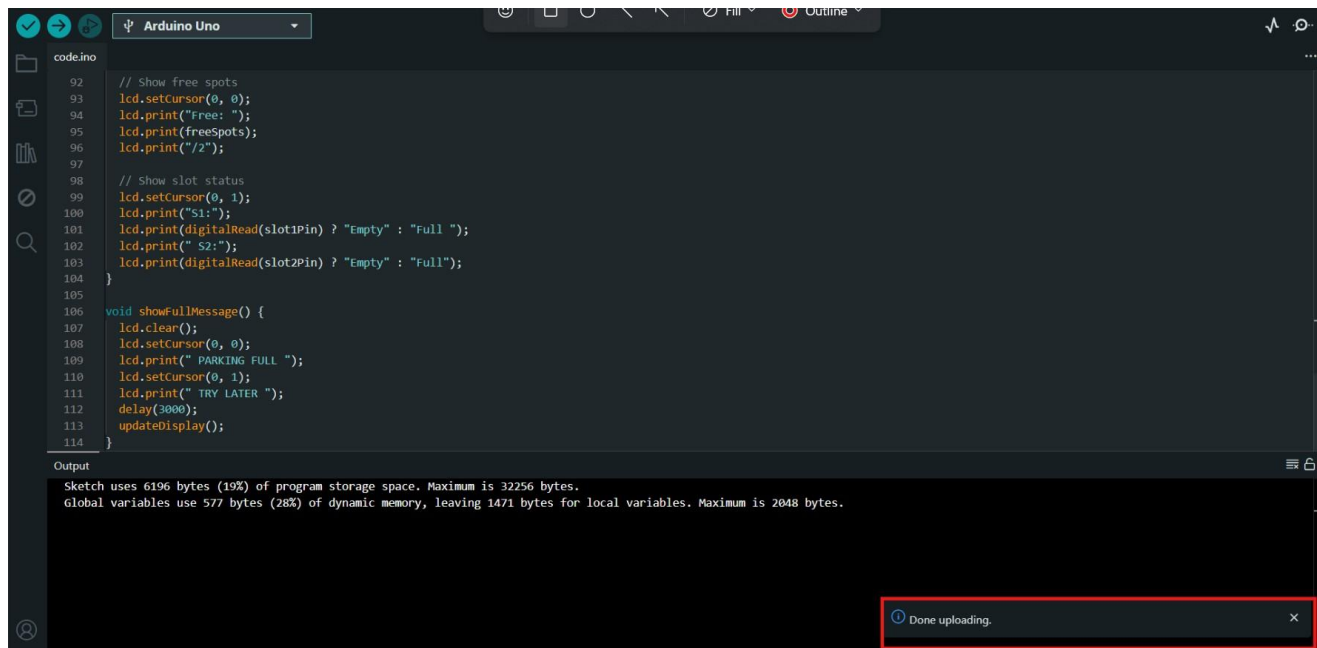


Figure 23: Successfully uploaded the code

Test 2: Check for exit and entry operability

Test No	2
Objective	To check whether the servo motor opens the barrier when a vehicle is at the entrance and exit.
Activity	To check the IR connectivity by placing an object (toy car) Infront of the exit and entrance gate.
Expected result	The gate should be able to open and close, allowing the vehicles to pass through easily.
Actual Result	The gate was easily able to allow access to the vehicle when detected by the IR sensor.
Conclusion	The test was Successful.

Table 8: Check for exit and entry operability

The servo motor opens when the entrance IR detects a movement.

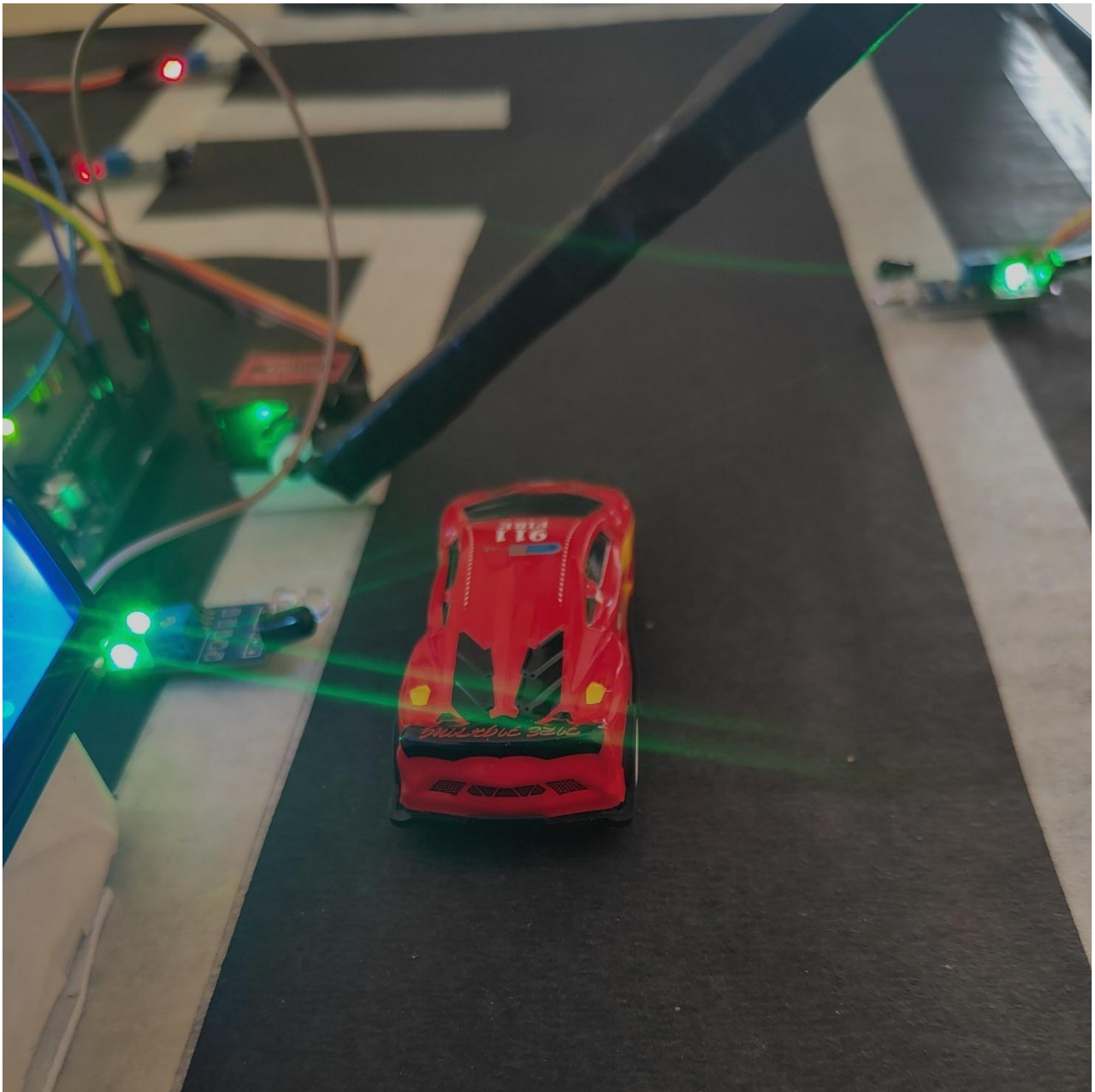


Figure 24: Barrier opening at entry

The servo motor opens when the entrance IR detects a movement.

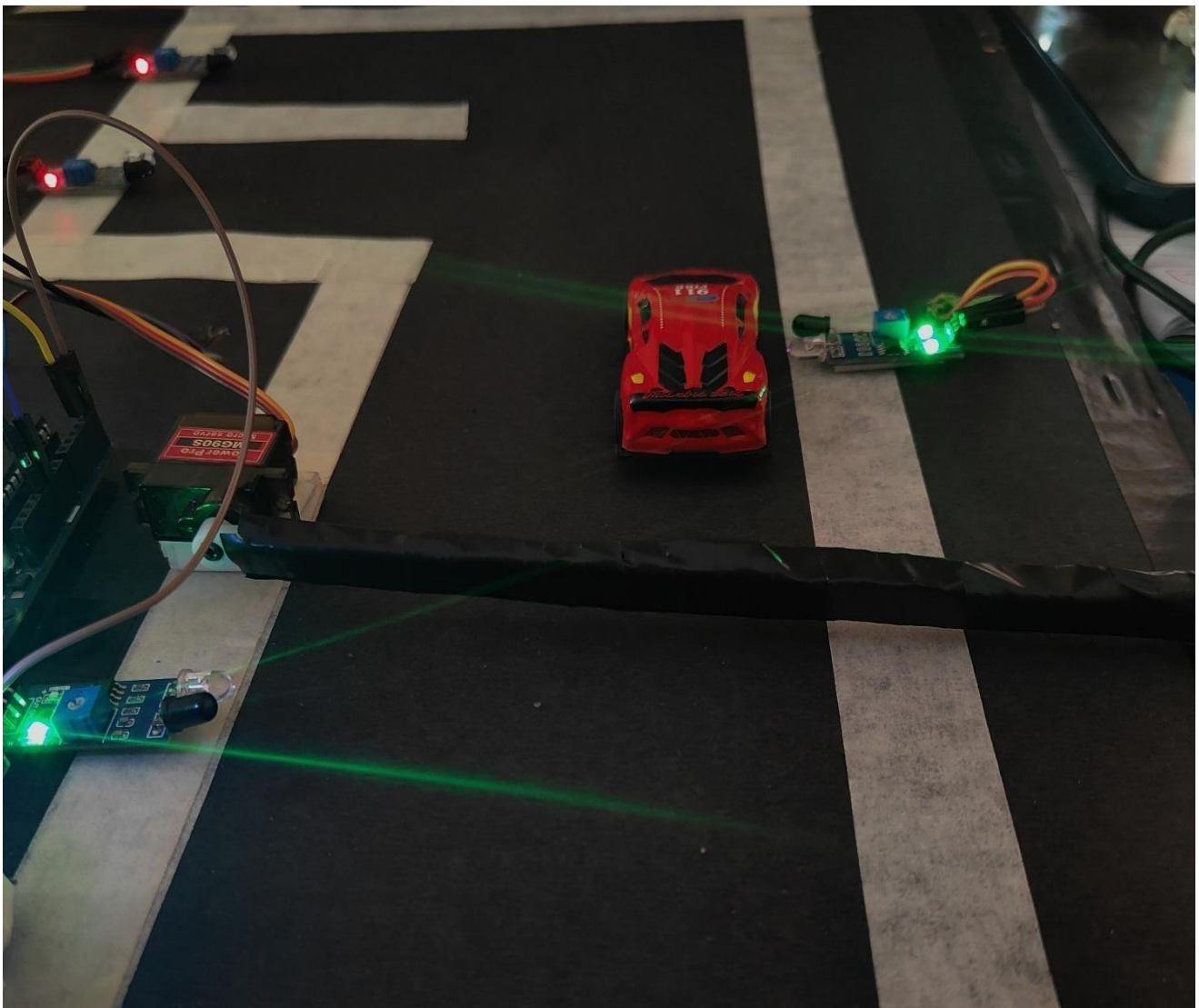


Figure 25: Barrier closes at entry.

The servo motor opens when the Exit IR detects a movement.

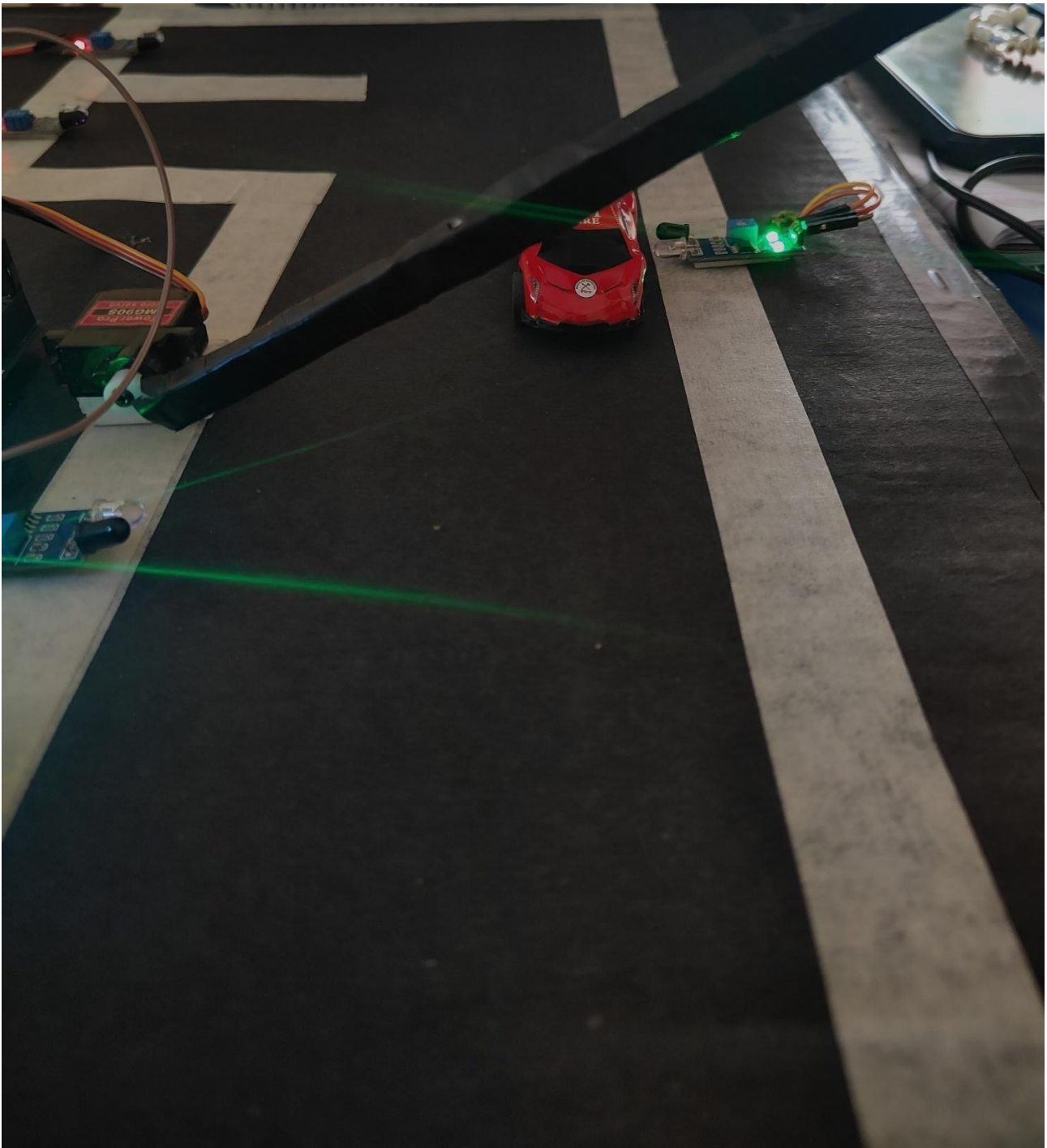


Figure 26: Barrier opening at exit

The servo motor close when the Exit IR detects a movement.

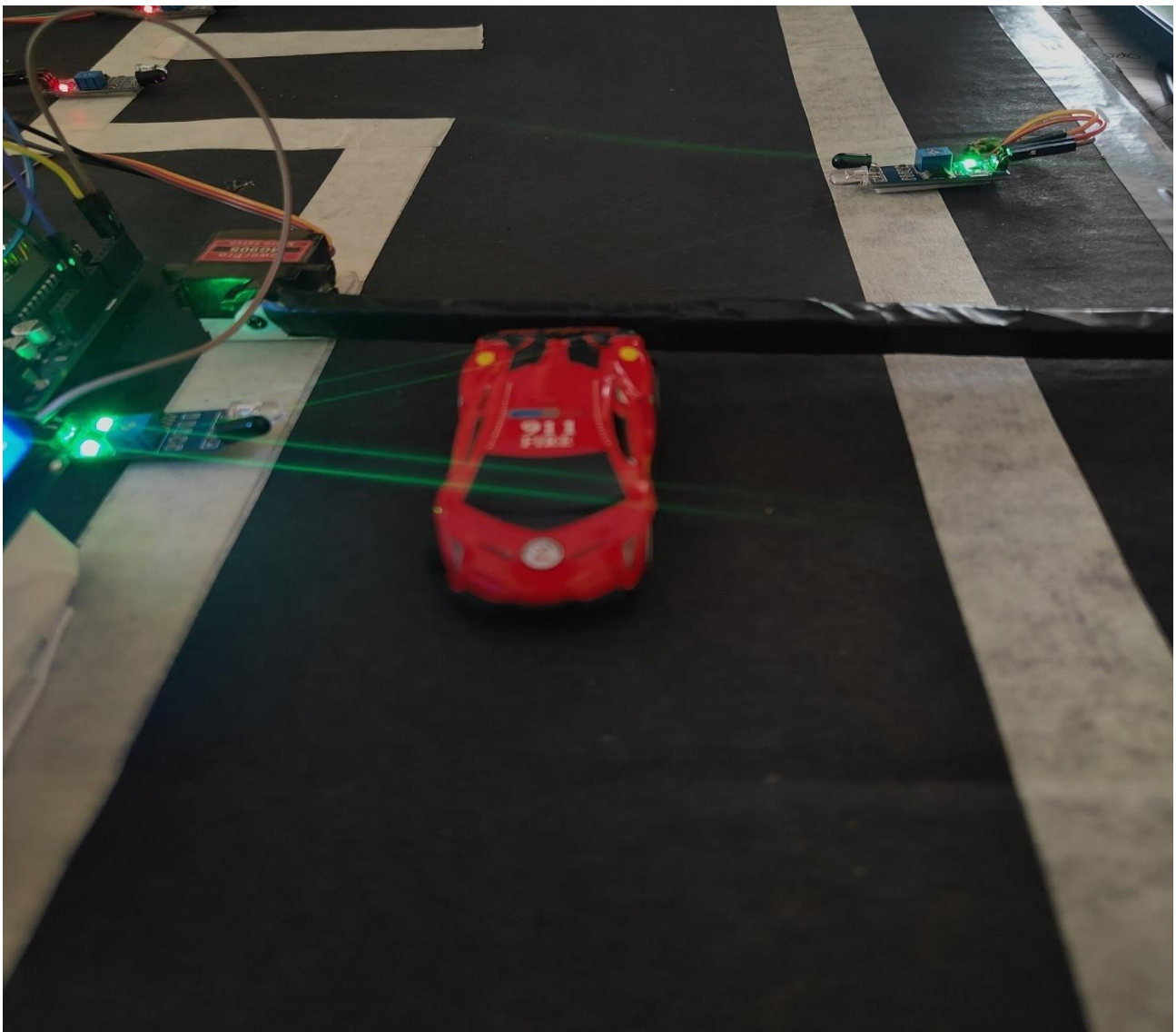
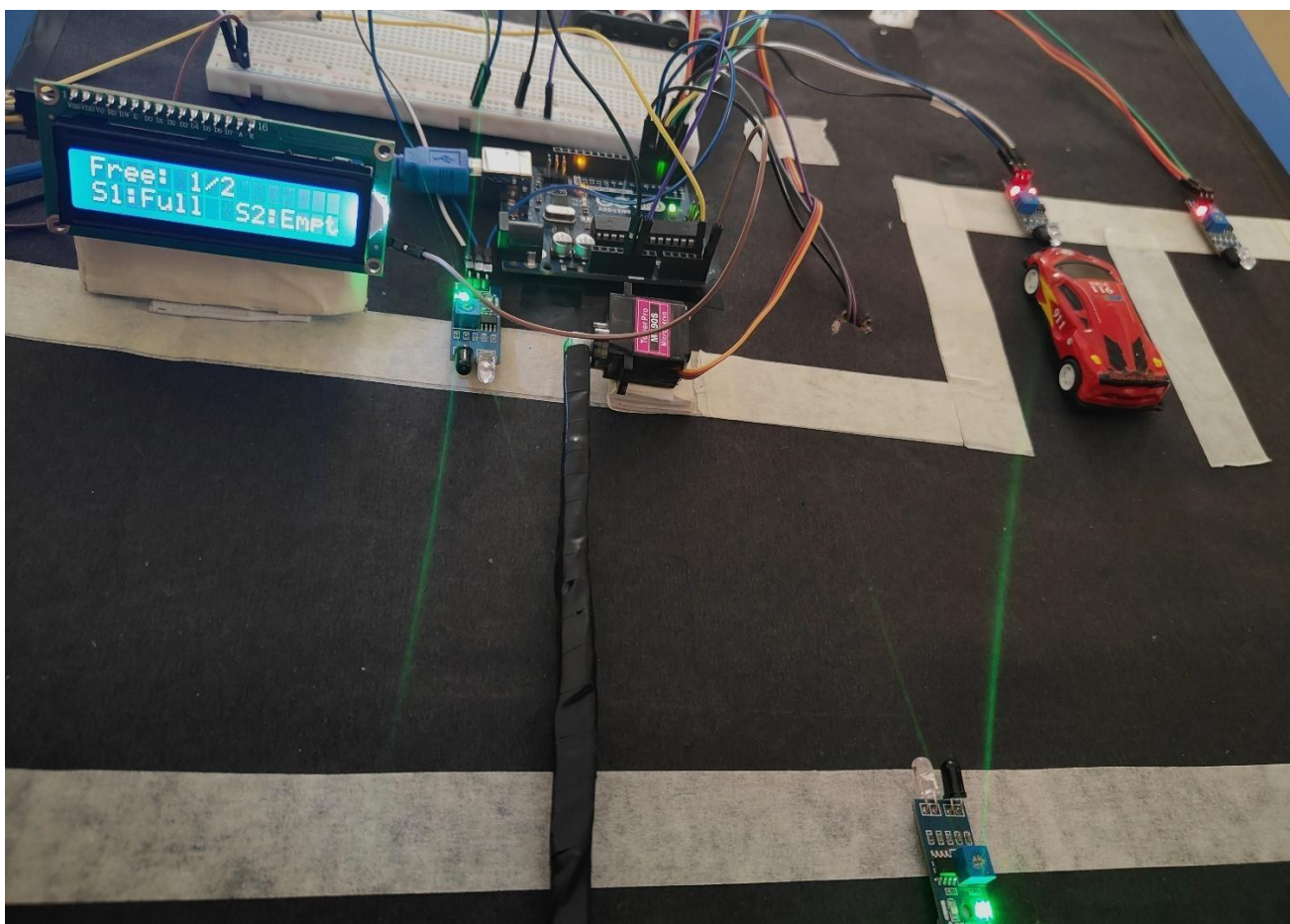


Figure 27: Barrier closes at exit.

Test 3: Check for Slot 1 and Slot 2 status

Test No	3
Objective	To check if the LED displays show the available slots when vehicles are parked in that slot.
Activity	To place the vehicle Infront of the each S1 and S2 and both parking slot to detect if the spot will be marked as empty or full according to the occupancy in the LED display.
Expected result	The LED should show the booked slot as full and the empty slot as empty.
Actual Result	The LED display showed the actual status of the slots in real time.
Conclusion	The test was Successful.

Table 9: Check for Slot 1 and Slot 2 status**Figure 28: Checking when S1 is occupied**

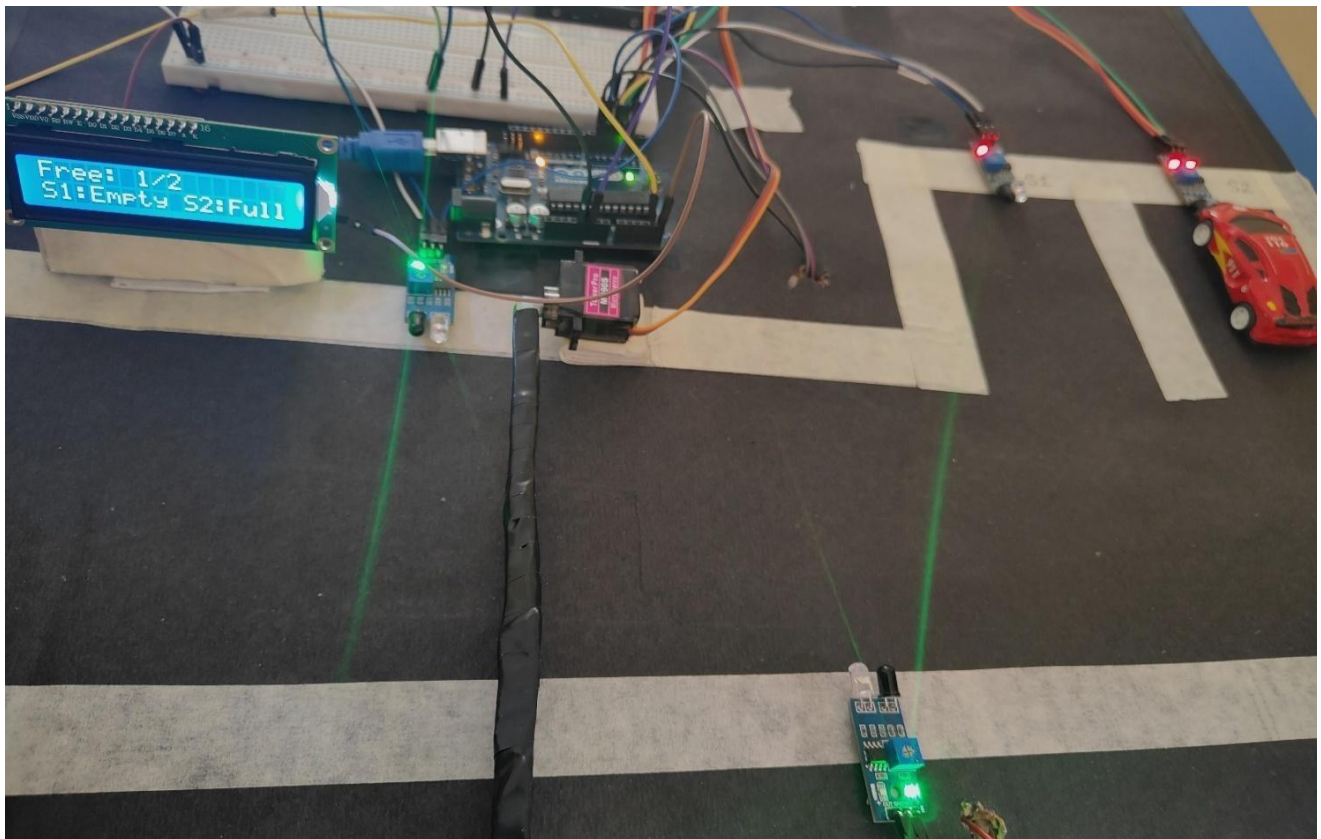


Figure 29: Checking when S2 is occupied

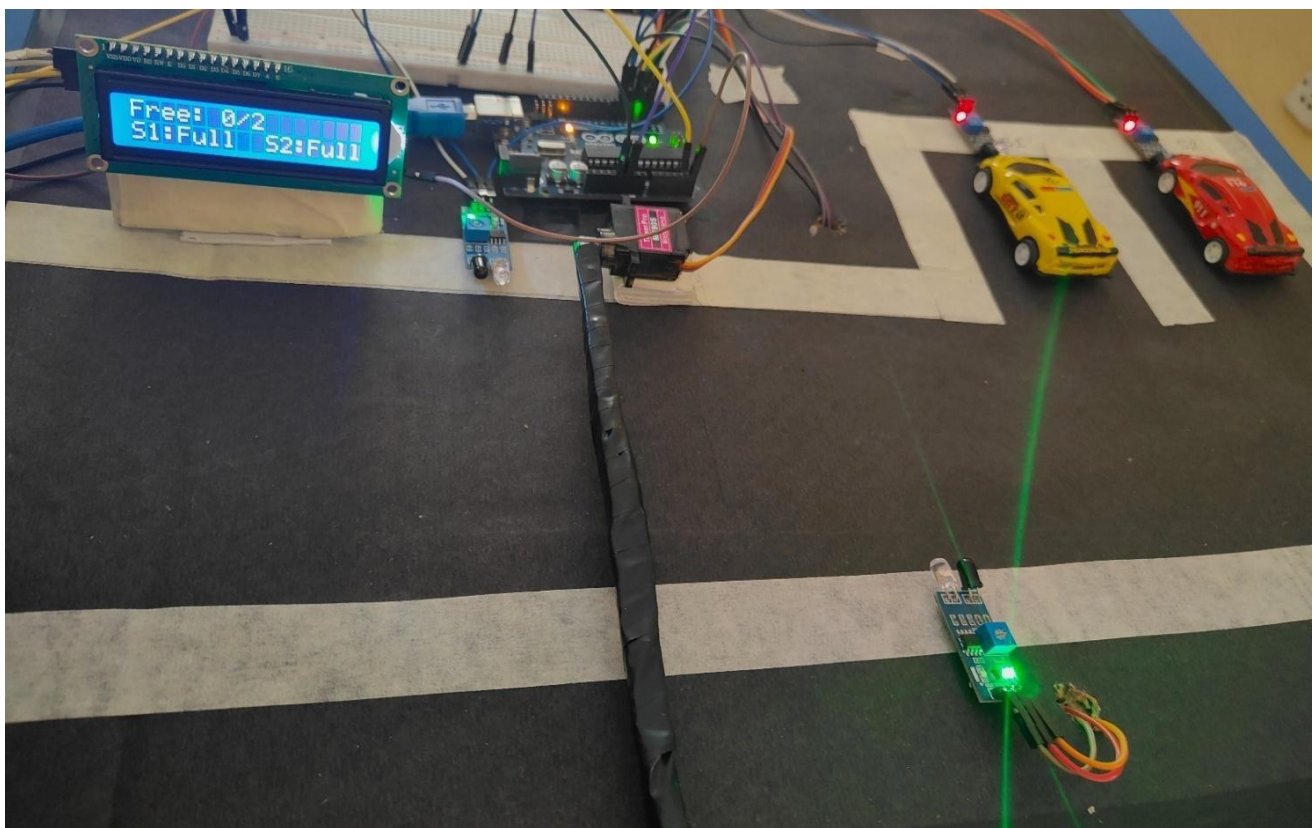
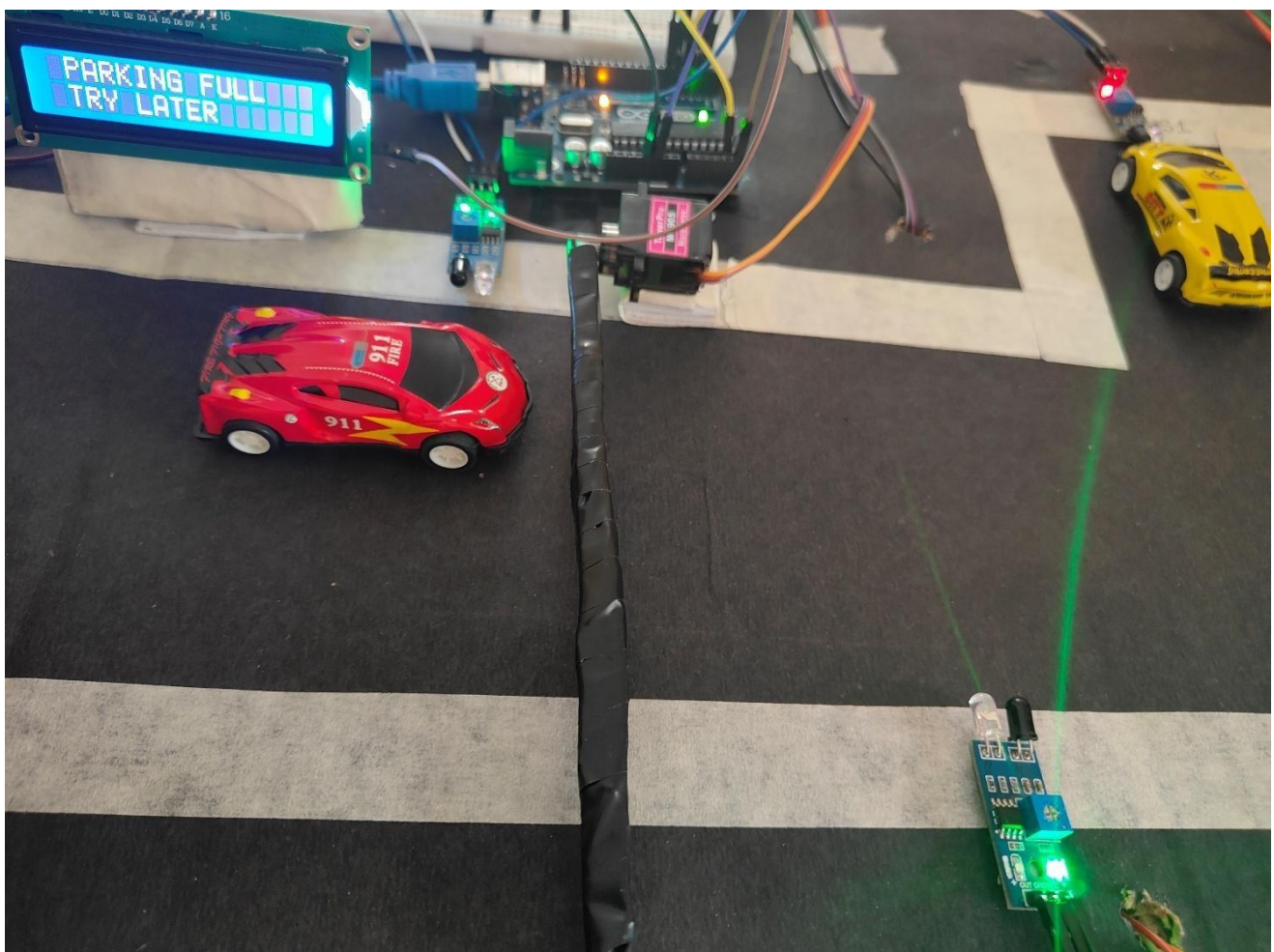


Figure 30: Checking when both S1 and S2 is occupied

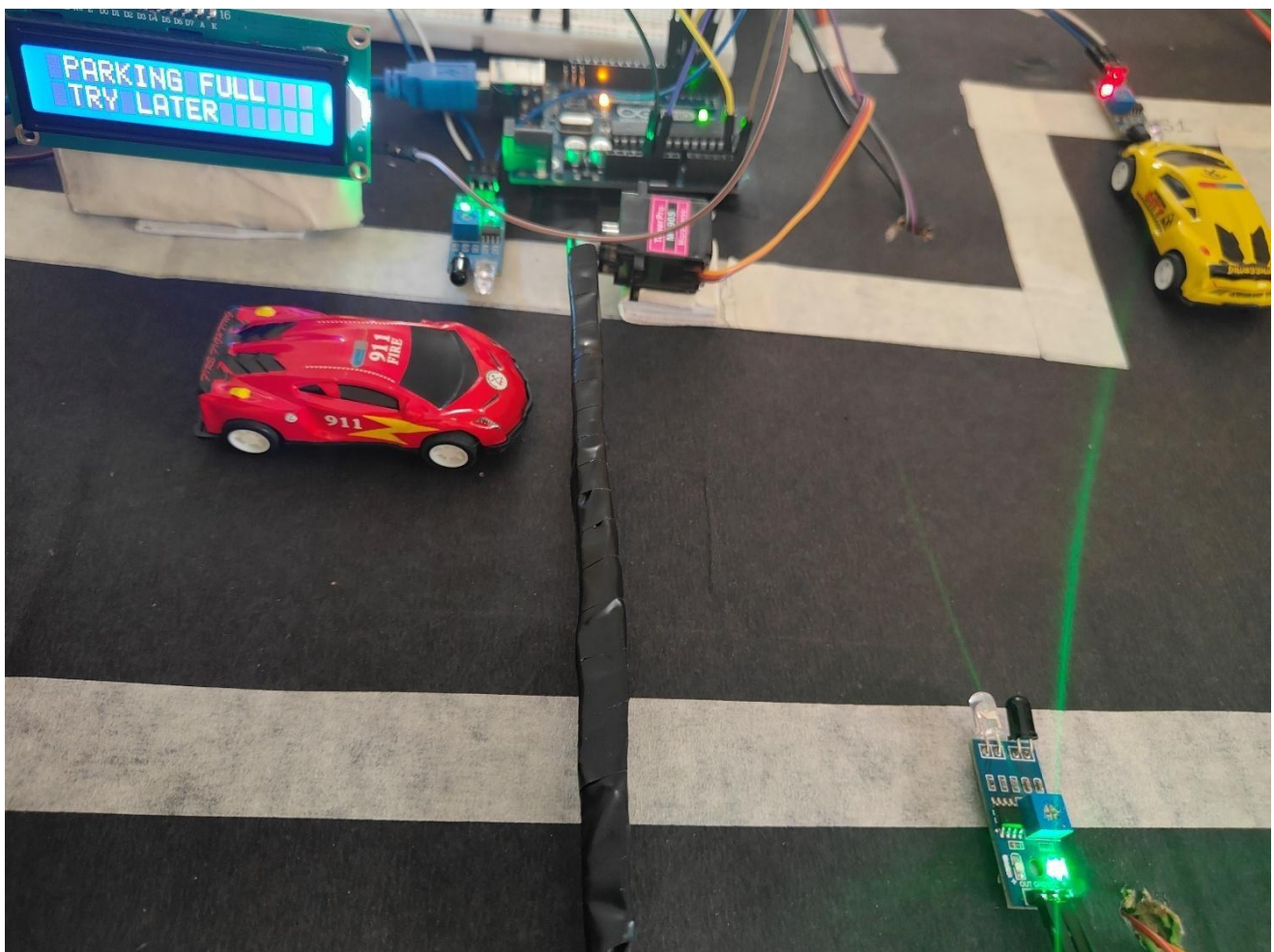
Test 4: Check for the try again message when parking is full.

Test No	4
Objective	If the slot counter is 0/2 then the display should show a try again later message.
Activity	Place a car in front of the IR then the display should show 0/2 (parking space to its maximum occupancy)
Expected result	The display should show a message showing message to try again.
Actual Result	The display showed stating 'PARKING FULL TRY LATER'
Conclusion	The test was Successful.

Table 10: Check for the try again message when parking is full**Figure 31: Parking full message when at maximum occupancy**

Test 5: Check if barrier opens when the slots are full but the occupancy free status.

Test No	5
Objective	Check if the barrier opens when the occupancy is not at its maximum.
Activity	Place toy car at the IR when the S1 and S2 are marked empty but the occupancy shoes 0/2 at the display.
Expected result	The barrier should open and let the vehicle pass if the Slots are empty
Actual Result	The barrier didn't open, and the display showed Try again prompt.
Conclusion	The test was Unsuccessful.

Table 11: Check if barrier opens when the slots are full but the occupancy free status**Figure 32: Barrier didn't open**

Test 6: Check if occupancy goes below the available number of slot count

Test No	6
Objective	Check if the number of available slots decreases more than the number of available slots (0) that are in the parking lot.
Activity	Place more than one device in front of the exit IR when the slot counter is already 0/2.
Expected result	The slot counter should not go below the threshold set by the limiter in the code (in this case the minimum limit is 0)
Actual Result	The number of slots was never shown below 0.
Conclusion	The test was Successful.

Table 12: Check for the minimum number of Slots.

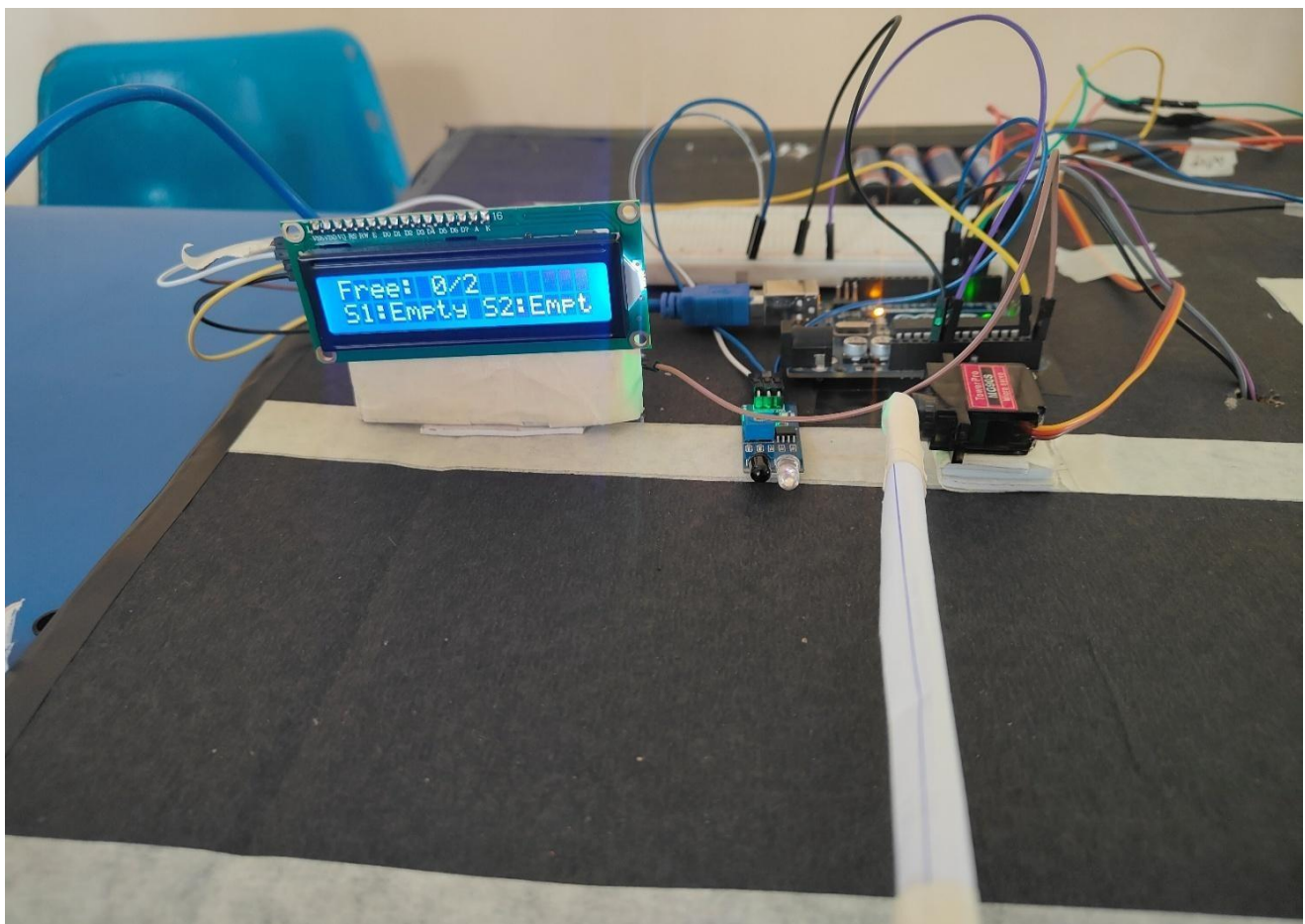


Figure 33: Slot counter at its maximum occupancy before

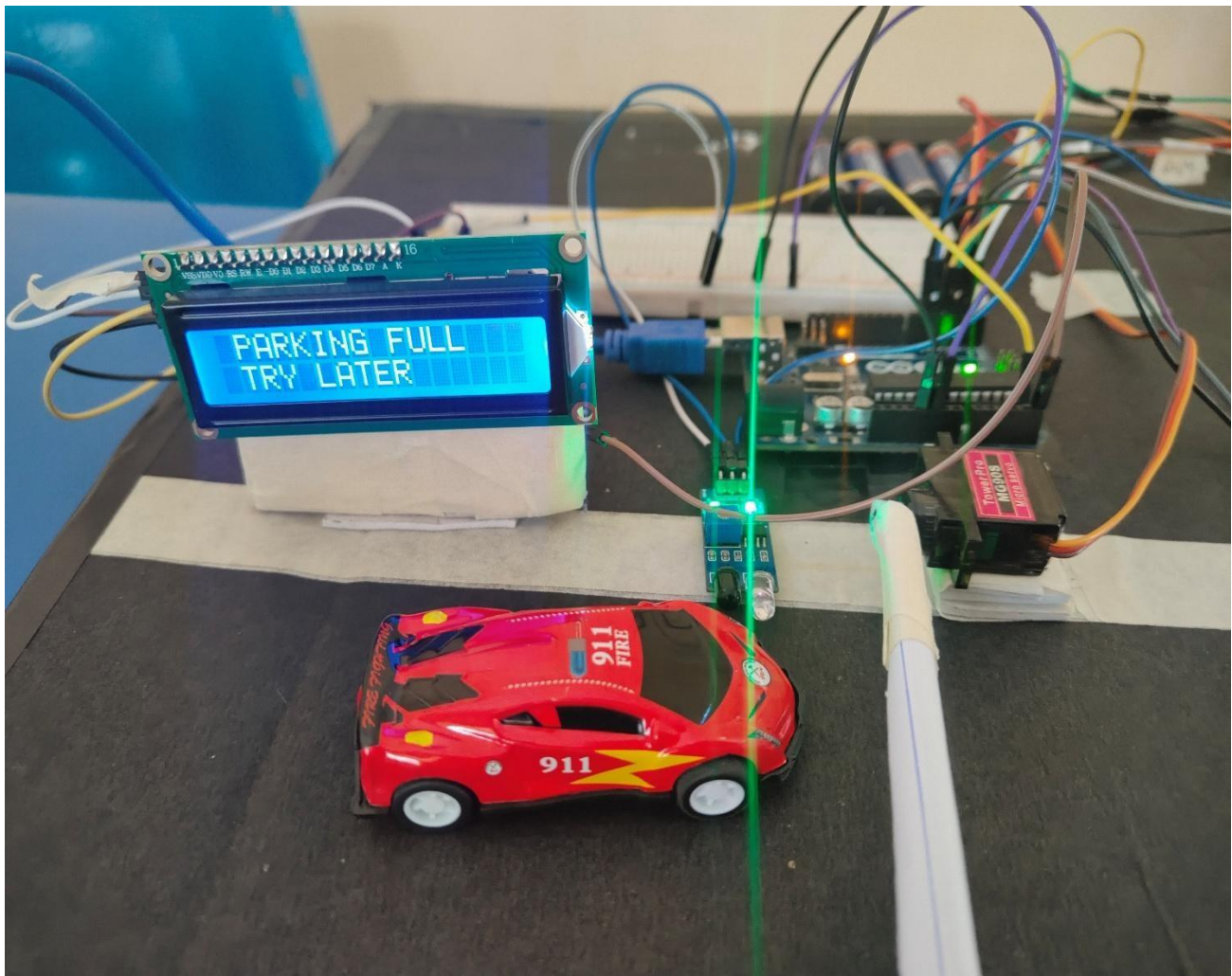


Figure 34: Slot counter at its maximum occupancy after

Test 7: Check if occupancy goes above the available number of slot count.

Test No	7
Objective	Check if the barrier opens when the occupancy is not at its maximum.
Activity	Place more than one device in front of the entry IR when the slot counter is already 2/2.
Expected result	The slot counter should not go above the threshold set by the limiter in the code (in this case the maximum limit is 2)
Actual Result	The number of slots was never shown above 2.
Conclusion	The test was Successful.

Table 13: Check for the minimum number of Slots

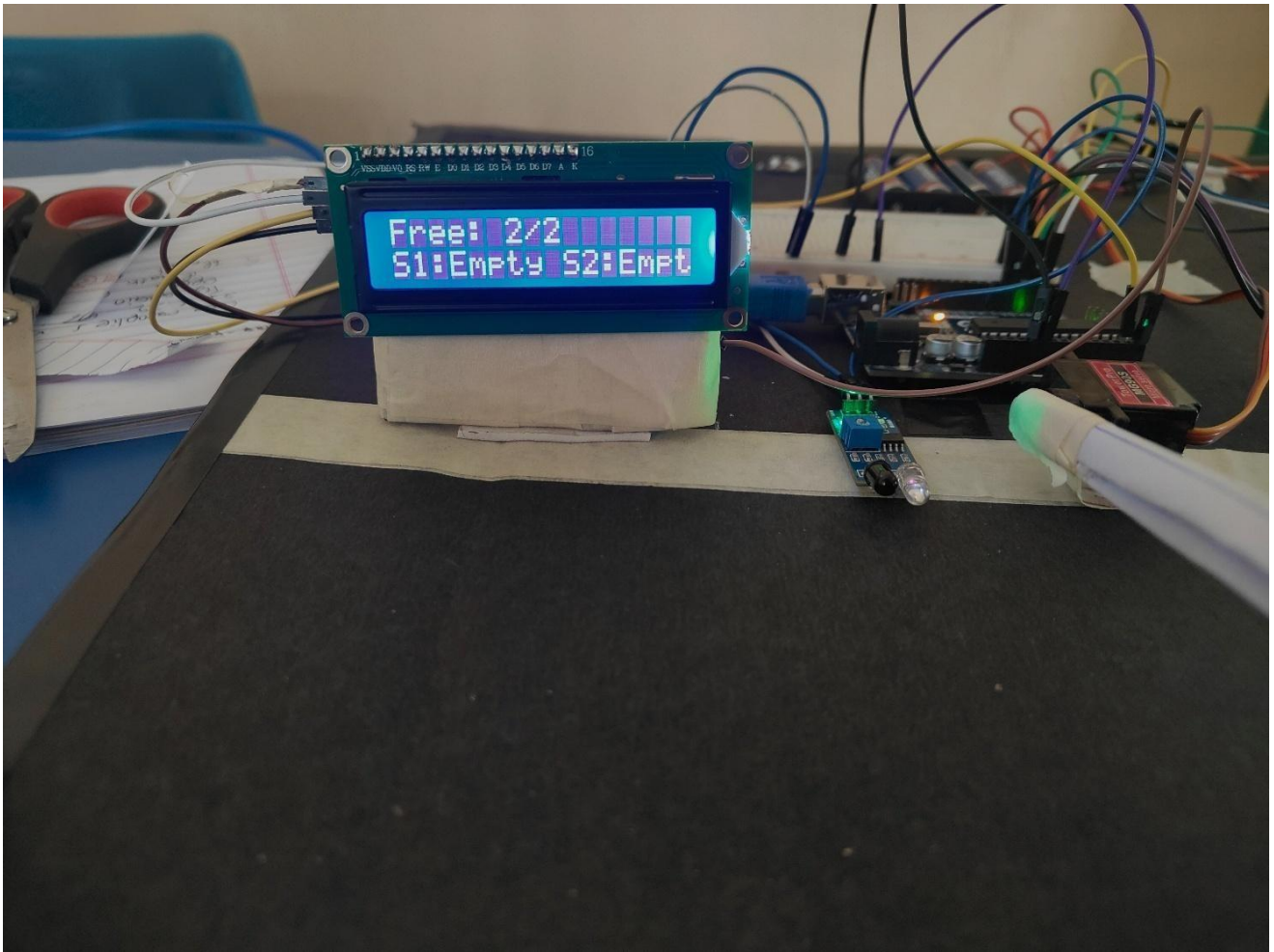


Figure 35: Slot Counter at its minimum occupancy before

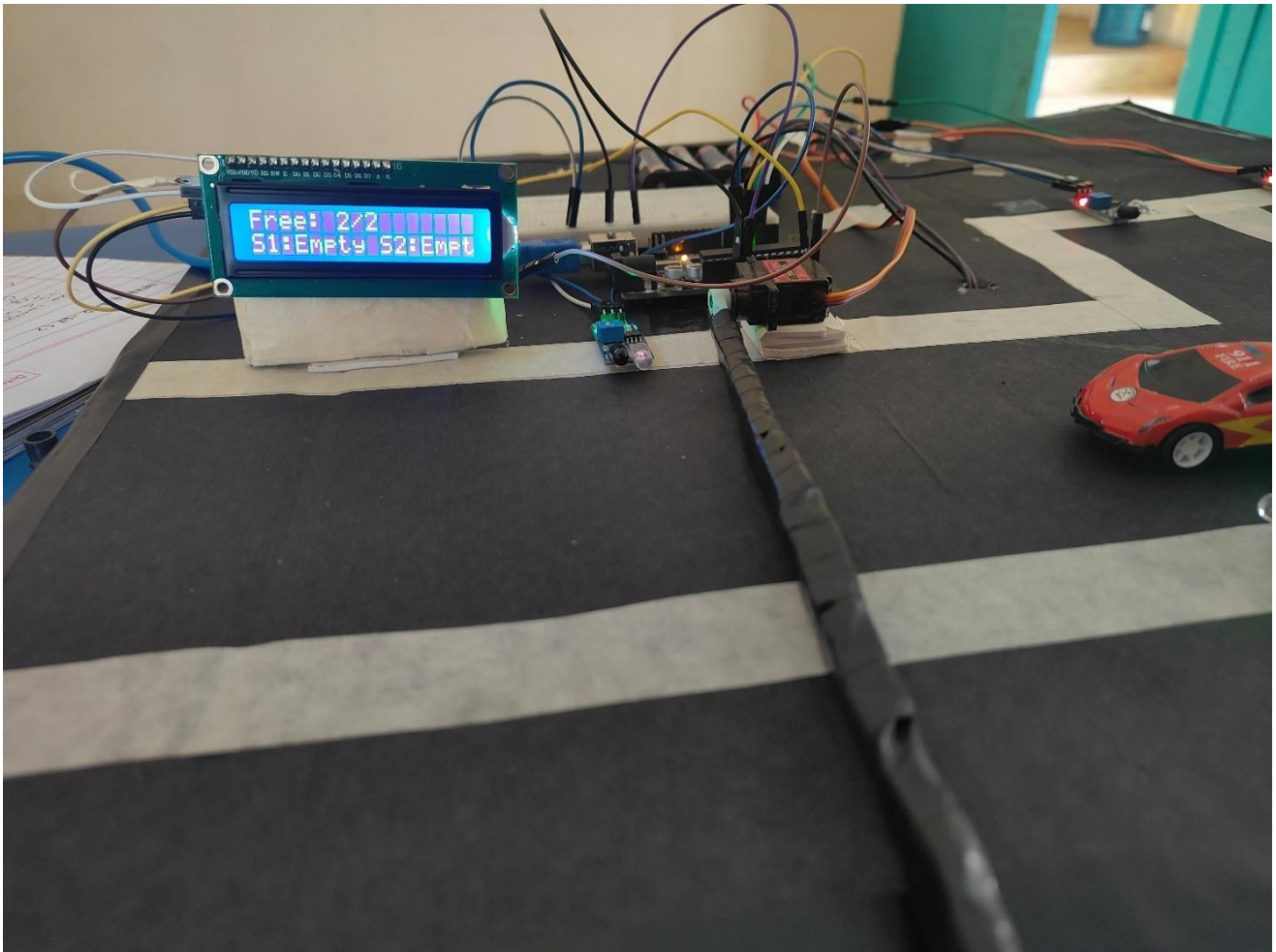


Figure 36: Slot Counter at minimum occupancy after

5. Future Works

This prototype is fully functional and practically possible to implement in real world. However, to make it more efficient, sustainable and scalable, many aspects can be added in the future so that it will be user and environment friendly and technologically advanced in the urban needs as well as changing smart city infrastructures. So, below is the example of systematic ways that can be made further in this prototype:

5.1 Sustainable Smart Parking System

Traditional parking systems contribute to significant environmental damage; in the U.S. alone, they lead to the consumption of 3.1 billion gallons of gasoline and 8.4 billion kilograms of carbon emissions annually, with 30% of fuel wasted by drivers searching for parking. Automated parking systems reduce congestion by guiding drivers in real time, decreasing fuel consumption and carbon emissions. Organizations that invest in these systems achieve long-term environmental and financial benefits, making modern parking technology a key opportunity for sustainability. (Conure, 2023)

Examples:

- Deploy solar-powered parking sensors to minimize dependence on the traditional electrical grid and lower energy costs.
- Establish designated “green zones” for electric and hybrid vehicles by prioritizing their access.

Implementation Strategy:

- Power IoT infrastructure using renewable energy sources.
- Introduce a carbon footprint tracker for vehicles utilizing the facility.
- Provide automated parking suggestions based on environmental preferences.

5.2 Electric Vehicle (EV) Support

EV adoption is becoming more popular every day, with global sales expected to reach nearly 15.9 million by 2025. To keep up, charging stations are almost everywhere and soon about to exceed 540,000 in the U.S., while in the UK there were nearly 80,000 charging points across over 39,000 locations as of April 2025. So, to encourage this adoption more in the future, smart parking system with smart charging technology is effective. (KORE, 2025)

Examples:

- Dynamically allocate charging stations based on parking duration and real time demand.
- Adjust charging fees according to energy availability and usage patterns.

Implementation Strategy:

- Install smart EV chargers that communicate with the parking management system.

- Develop mobile applications for reserving charging slots in real time.
- Motivate sustainable transport by offering priority zones for EVs.

5.3 Enhanced Security & Surveillance

1.2 million parking lot accidents occur annually in the United States accounting for about 14% of all vehicle crashes in urban areas making parking lots one of the most common yet often overlooked hotspots for vehicular collisions and injuries (Lindner, 2025). Criminals often target these zones due to the higher likelihood of encountering potential victims. So, equipment such as live monitoring system, two-way audio communication and virtual security personnel are converting an inactive surveillance to an active crime prevention (Trassir, 2024). Hence, parking lot surveillance is a key aspect of modern security measures, addressing the increasing demand for safety of vehicles and user trust.

Examples:

- Automatic Detection of Parking Violations.
- Implement AI-powered surveillance for real-time anomaly detection.
- Utilize biometric or RFID-based vehicle access control.

Implementation Strategy:

- Detect and send immediate alerts to improperly parked vehicles; e.g., in handicapped spots, blocking traffic lanes through AI and sensors.
- Connect surveillance feeds to a cloud-based security platform.
- Enable automated alerts for unauthorized access or suspicious activity.
- Encrypt all data transmissions to safeguard user privacy.

5.4 Integration with Autonomous Vehicles

With the autonomous vehicles (AVs) becoming a reality, the parking industry has to transform to support the new vehicles. Currently, the average privately owned vehicle in the United States is only utilised 5% of the time, while shared vehicles are estimated to have a utilisation rate closer to 40%. This paradigm shift creates an opportunity for the transformation of parking lots into service centres and waiting areas until AV is requested by user. Parking lot owners and investors may also benefit by utilizing the additional capacity brought by AVs hence generating more revenues due to the efficient usage of space (Anita Mauchan, 2018).

Examples:

- Develop self-parking capabilities that allow AVs to park in designated smart parking spaces without human input.

- Utilize AV data to predict demand and availability, improving efficiency in parking allocation.

Implementation Strategy:

- Install docking stations for AVs where they can recharge, park, and be retrieved automatically.
- Ensure that parking management systems are compatible with AV communication protocols for seamless interaction.
- Implement predictive analytics to optimize parking flow and ensure smooth operation with AVs.

6. Conclusion

This Smart parking System can demonstrate how and IoT implemented technology can be used to address and solve the problems of urban areas such as challenges created by heavy traffic and large population areas and use of automation to tackle those issues. The traditional parking method is most easy to operate but comes with multiple hidden problems in the long run like congestion, manual errors and inefficient space utilization. While also being difficult to implement in high traffic and urban areas. This IoT project focused on reducing the congestion and errors created by traditional parking system while reducing time on search for vacant spot through smart control unit that displays the slot status at the entry which will allow the driver to know about the available location if there are any and park accordingly.

The IoT based prototype project uses multiple IR for slot detection and updating the LCD for occupancy in the parking slot and use of actuators like servo motor to open and close the barrier depending upon the availability of the slots and the led display to how a vehicle about the status beforehand entering the parking lot looking for an available spot. To help with the project multiple diagrams such as block diagram, system architecture, circuit diagram, schematic diagram and flowchart were made with the help of web application such as tinker cad and draw.io. After getting all the relevant information on what problems can occur during the development phases and conducting virtual testing using the inbuilt feature of the circuit maker, the prototype was able to be developed further by connecting the input, output device with the controlling unit and the power source.

The prototype was tested on 7 different aspects including code, functionality of the prototype and logic of the code. Even there were minor hiccups while in the development of the code the prototype is completed but there are rooms for further future improvements and expansions like use of AI for Dynamic pricing, implementation of chargers for the EV 's, use more secure surveillance for better security, integration with autonomous vehicles & Payments, use of mobile application and much more. This project aims to solve the common problems brought by the traditional parking systems such as unnecessary wastage of time while searching for available spots and traffic congestion within the parking spot. The prototype helped with both problems by helping with the easy detection of available parking spaces and reducing human involvement in searching the available spots

The project can make a mark in IoT related development projects in urban areas and proves that low-cost technology can be used to develop impactful and scalable solutions and overall, this project is a major step in the path to smarter and more efficient urban infrastructure.

7. References

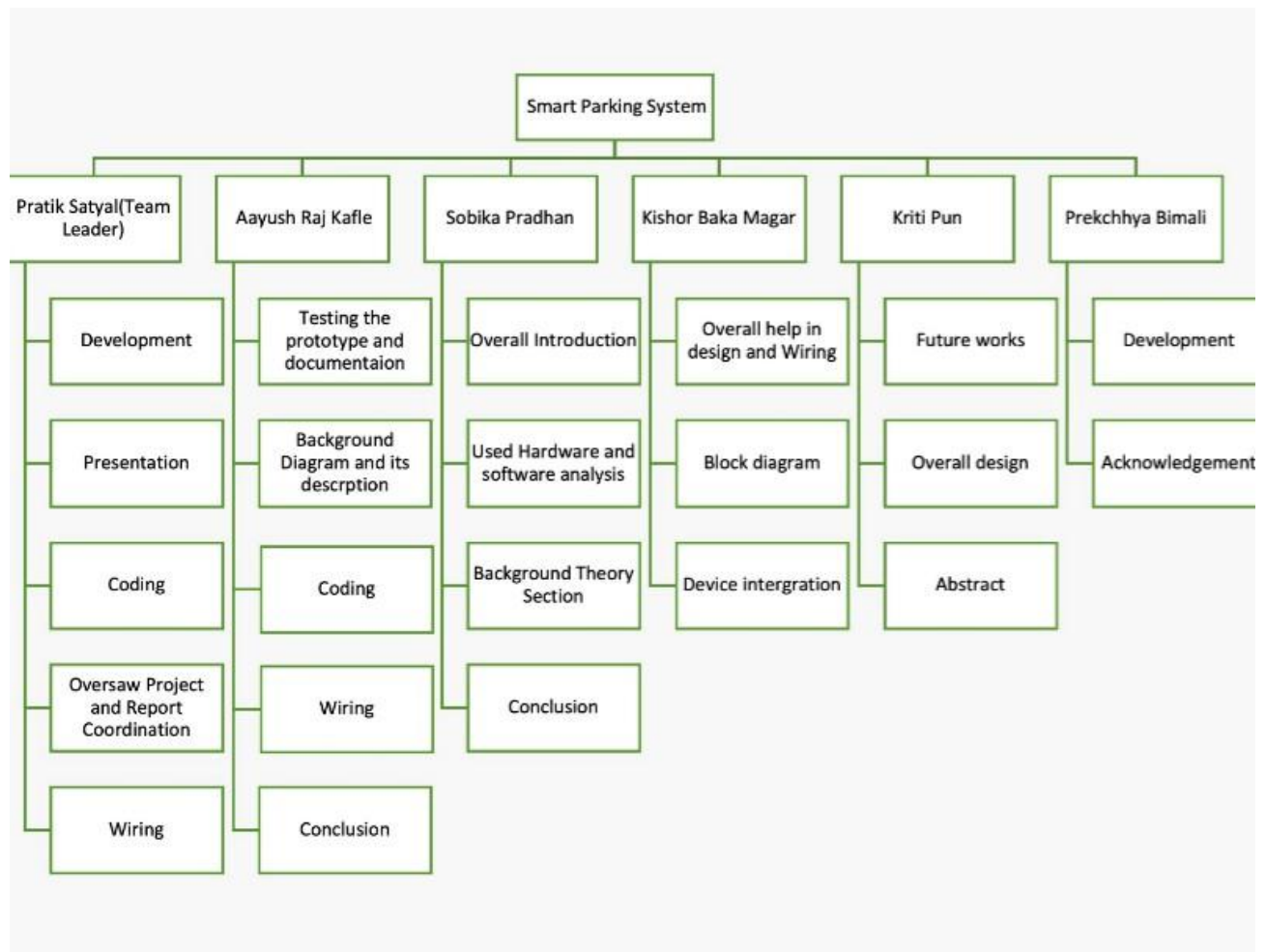
- [1]. Anderson, W., 2012. *How to Install the Arduino IDE for Windows*, s.l.: s.n.
- [2]. Anita Mauchan, A. M. A. H., 2018. *How will autonomous vehicles change parking?*, s.l.: s.n.
- [3]. Apoorve, 2025. *What is a Servo Motor?*. [Online] Available at: <https://circuitdigest.com/article/servo-motor-working-and-basics> [Accessed 12 05 2025].
- [4]. AutoDesk, 2015. *File:Logo-tinkercad-wordmark.svg*, s.l.: s.n.
- [5]. Barbosa, A., Vale, I. & Alvarenga, D., 2024. design, The use of Tinkercad and 3D printing in interdisciplinary STEAM education: A focus on engineering. *STEM Education*, Volume 4, pp. 222--246.
- [6]. Conure, 2023. *The Top 5 Parking Solutions to Support a Sustainable Future*, s.l.: s.n.
- [7]. Crowell, G., 2019. *CircuitBoard*. [Online] Available at: <https://www.circuitbread.com/ee-faq/what-is-a-breadboard> [Accessed 03 05 2025].
- [8]. element14, 2025. *Jumper Wire Kit, Male to Female, Multi-Coloured, 200 mm, 0.1" Dupont Connector, 0.2 mm²*. [Online] Available at: <https://au.element14.com/multicomp-pro/mp006289/jumper-wire-kit-male-to-female/dp/3617778> [Accessed 12 05 2025].
- [9]. Firoozian, R., 2014. *Servo motors and industrial control theory*. s.l.:Springer.
- [10]. Hiner, S., 2023. *Driving Sustainability: The Transformative Impact of Smart Parking Solutions on Our Environment*, s.l.: s.n.
- [11]. INRIX Research, 2017. *INRIX 2017 Global Traffic Scorecard: Parking Pain in America*, Kirkland, WA: INRIX.
- [12]. Kaplan, J. & Rabelo, L., 2024. Preliminary Studies to Bridge the Gap: Leveraging Informal Software Architecture Artifacts for Structured Model Creation. *Information*, Volume 15, p. 642.
- [13]. KORE, 2025. *10 EV Charging Statistics You Should Know*, s.l.: s.n.

- [14]. Kumar, R., Roopa, A., Sathiya, D. P. & others, 2015. Arduino ATMEGA-328 microcontroller. *Int. J. Innov. Res. Electr. Electron. Instrum. Control Eng*, Volume 3, pp. 27--29.
- [15]. Lall, P. & Pecht, M., 2018. Interconnections and Connectors. In: *Handbook of Electronic Package Design*. s.l.:CRC press, pp. 239--291.
- [16]. Lindner, J., 2025. *Parking Lot Accident Statistics*, s.l.: Gitnux.
- [17]. link, B. C. a., 2025. *draw.io*, s.l.: Pinterest.
- [18]. Perwej, Y. et al., 2019. The internet of things (IoT) and its application domains. *International Journal of Computer Applications*, Volume 975, p. 182.
- [19]. Rogalski, A. & Chrzanowski, K., 2017. Infrared devices and techniques. In: *Handbook of optoelectronics*. s.l.:CRC Press, pp. 633--686.
- [20]. Smart Life, 2022. *Parking System*. [Online]
Available at: <https://www.smartlifeeg.com/parking-system>
[Accessed 2025].
- [21]. Spiceman, 2022. *How to use a Breadboard*. [Online]
Available at: <https://spiceman.net/bread-board/>
[Accessed 12 05 2025].
- [22]. Teja, R., 2024. *Arduino UNO Pinout, Specifications, Board Layout, Pin Description*. [Art] (electronicshub).
- [23]. TheAseanPost , 2019. *Smart cities need smart parking*. [Online]
Available at: <https://theaseanpost.com/article/smart-cities-need-smart-parking>
[Accessed 13 05 2025].
- [24]. ThinkRobotics, 2025. *What is Arduino IDE? A Complete Guide for Beginners*. [Online]
Available at: <https://thinkrobotics.com/blogs/learn/what-is-arduino-ide-a-complete-guide-for-beginners>
[Accessed 03 05 2025].
- [25]. Trassir, 2024. *Revolutionizing Parking Lot Security: Advanced Surveillance Strategies for Safer Spaces*, s.l.: s.n.

- [26]. Watson, D., 2023. *Interface Sharp Infrared Distance Measurement Sensor*. [Online] Available at: https://www.theengineeringprojects.com/2023/04/interface-sharp-infrared-distance-measurement-sensor-with-raspberry-pi-4.html#google_vignette [Accessed 12 05 2025].

8. Appendix

8.1 Individual contribution Plan



8.2 Code used for Smart Parking System

[Skip to content](#)

[Using Gmail with screen readers](#)

[Conversations](#)

[37% of 15 GB used](#)

[Terms](#) · [Privacy](#) · [Program Policies](#)

[Last account activity: 53 minutes ago](#)

[Details](#)

```
#include <Wire.h>
```

```
#include <LiquidCrystal_I2C.h>
```

```
#include <Servo.h>
```

```
// LCD setup (0x27 is common I2C address for 16x2 LCD)
```

```
LiquidCrystal_I2C lcd(0x27, 16, 2);
```

```
// Servo setup for gate control
```

```
Servo gate;
```

```
const int gatePin = 4;
```

```
// IR sensor pins
```

```
const int entryPin = 5;  // Entry sensor
```

```
const int exitPin = 6;   // Exit sensor
```

```
const int slot1Pin = 2;  // Parking slot 1

const int slot2Pin = 3;  // Parking slot 2


// Parking system variables

int freeSpots = 2;      // Start with 2 free spots

bool gateMoving = false; // To prevent multiple gate operations

unsigned long gateTimer = 0;


void setup() {

  Serial.begin(9600);


  // Initialize LCD

  lcd.init();

  lcd.backlight();


  // Setup servo

  gate.attach(gatePin);

  closeGate(); // Start with gate closed


  // Setup sensors

  pinMode(entryPin, INPUT_PULLUP);

  pinMode(exitPin, INPUT_PULLUP);

  pinMode(slot1Pin, INPUT_PULLUP);

  pinMode(slot2Pin, INPUT_PULLUP);
```

```
// Show welcome message

lcd.setCursor(0, 0);

lcd.print(" SMART PARKING ");

lcd.setCursor(0, 1);

lcd.print("  SYSTEM  ");

delay(2000);


updateDisplay();

}


void loop() {

  // Check if car is entering

  if (digitalRead(entryPin) == LOW && !gateMoving) {

    if (freeSpots > 0) {

      openGate();

      freeSpots--;

      updateDisplay();

    } else {

      showFullMessage();

    }

  }


  // Check if car is exiting
```

```
if (digitalRead(exitPin) == LOW && !gateMoving) {  
    openGate();  
    freeSpots = min(2, freeSpots + 1); // Never exceed 2 spots  
    updateDisplay();  
}  
  
// Handle automatic gate closing  
if (gateMoving && millis() - gateTimer >= 5000) {  
    closeGate();  
    gateMoving = false;  
}  
  
// Update display every second  
static unsigned long lastUpdate = 0;  
if (millis() - lastUpdate >= 1000) {  
    updateDisplay();  
    lastUpdate = millis();  
}  
}  
  
void openGate() {  
    gate.write(180);    // Open gate (180° position)  
    gateMoving = true;  
    gateTimer = millis();  
}
```

```
}
```

```
void closeGate() {  
    gate.write(90);    // Close gate (90° position)  
}
```

```
void updateDisplay() {  
    lcd.clear();  
  
    // Show free spots  
    lcd.setCursor(0, 0);  
    lcd.print("Free: ");  
    lcd.print(freeSpots);  
    lcd.print("/2");  
  
    // Show slot status  
    lcd.setCursor(0, 1);  
    lcd.print("S1:");  
    lcd.print(digitalRead(slot1Pin) == HIGH ? "Empty" : "Full ");  
    lcd.print(" S2:");  
    lcd.print(digitalRead(slot2Pin) == HIGH ? "Empty" : "Full");  
}
```

```
void showFullMessage() {
```

```
lcd.clear();  
  
lcd.setCursor(0, 0);  
  
lcd.print(" PARKING FULL ");  
  
lcd.setCursor(0, 1);  
  
lcd.print(" TRY LATER ");  
  
delay(3000);  
  
updateDisplay();  
  
}  
  
#include Wire.h.txt  
  
Displaying #include Wire.h.txt.
```