Project Based Learning Report on

# Automated Car Parking System using Sensors

Submitted in the partial fulfillment of the requirements for the Project based learning in Industrial IOT and ML

ELECTRONICS AND COMMUNICATION ENGINEERING

By

|  |  |  |
| --- | --- | --- |
| **Name of the student** | **PRN** | |
| Bhavya Jain | | 2114110458 | |
| Satyam Mishra | | 2214110605 | |
| Ram Prakash | | 2214110681 | |

Under the guidance of Course Incharge Prof. Dr. A.A Shinde

Department of Electronics & Communication Engineering Bharati Vidyapeeth

(Deemed to be University) College of Engineering,

Pune – 411043

Year- 2023 -2024

## Bharati Vidyapeeth (Deemed to be University) College of Engineering,

**Pune – 411043**

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

## CERTIFICATE

Certified that the Project Based Learning report entitled, “Automated Car Parking System using Sensors” is a bonafied work done by

|  |  |
| --- | --- |
| **Name of the student** | **PRN** |
| **Bhavya Jain** | **2114110458** |
| **Satyam Mishra** | **2214110605** |
| **Ram Prakash** | **2214110681** |

in partial fulfillment of the requirements for the award of credits for Project Based Learning (PBL) in Industrial IOT and ML Course of Bachelor of Technology Semester VI, in Electronics and Communication Branch.

Date:

Dr. A.A Shinde Prof. Dr. A.A Shinde

Course Incharge Head of Department

# INDEX

|  |  |  |
| --- | --- | --- |
| **Sr. No** | **Content** | **Page No.** |
| 1. | Problem statement | 1 |
| 2. | Project Description | 1-2 |
| 3. | Software Used | 3 |
| 4. | Components Used | 4-6 |
| 5. | Flow Chart | 7 |
| 6. | Algorithm | 8 |
| 7. | Program | 9-12 |
| 8. | Circuit Diagram | 13 |
| 9. | Results | 14-18 |
| 10. | Conclusion | 19 |
| 11. | Course Outcome | 19 |
| 12. | References | 20 |

## Problem Statement

## Automated Car Parking System using Sensors

## Project Description

The Automated Parking System (APS) project aims to create a smart parking solution utilizing sensor technology for efficient management of parking spaces. The system will automate the entry and exit processes for vehicles, optimize parking space utilization, and enhance user convenience.

## Project Objectives: –

The objective of an Automated Parking System (APS) is to optimize parking space management, enhance user convenience, and improve operational efficiency through the integration of sensor technology, automation, and smart parking solutions. Key objectives include maximizing space utilization, streamlining entry and exit processes, enhancing user experience, increasing safety and security, reducing environmental impact, and supporting broader urban mobility initiatives.

## Project Goals: :–

## Efficiently utilize available parking space to accommodate more vehicles and minimize congestion.

## Streamline the parking process for users by automating entry and exit procedures, reducing wait.

## Implement measures to enhance the safety and security of vehicles and pedestrians within parking facilities, reducing the risk of accidents and theft.

* Reduce emissions and energy consumption by minimizing the time spent searching for parking spots and optimizing traffic flow within parking facilities.
* Ensure flexibility to accommodate various types of parking facilities and configurations.

## Components: -

* Arduino UNO
* IR Proximity Sensor
* Servo Motor
* 16x2 LCD I2C Display
* Jumpers
* Breadboard
* LED

**Key Points:-**

* The system uses IR sensors to detect the movement of vehicles and determine if a parking slot is empty or not.
* A 16x2 LCD is used to display the parking status, such as the number of available slots.
* The system uses a servo motor to block or allow the entrance of vehicles into the parking lot.
* The Arduino Uno microcontroller is used to program the system and control the components.
* The system is capable of finding available parking slots automatically and blocking the entrance if no slots are available.
* The code for the system includes functions to read the input from the IR sensors, display the parking status on the LCD, and control the servo motor.
* The system can be further enhanced by adding features such as automatic billing and security cameras.
* Overall, the automated parking system using Arduino Uno is an innovative solution to the increasing parking problems in urban areas. It provides a convenient and efficient way to manage parking spaces and can help reduce time and fuel waste associated with manual parking methods.

­­

**SOFTWARE USED**

The Arduino IDE (Integrated Development Environment) is an open-source software application used to write, compile, and upload programs (often called sketches) to Arduino-compatible microcontrollers. It provides a user-friendly interface for programming and interacting with Arduino boards, making it accessible to both beginners and experienced developers.

The Arduino Software (IDE) makes it easy to write code and upload it to the board offline. We recommend it for users with poor or no internet connection. This software can be used with any Arduino board.

**Key features of the Arduino IDE include:-**

1. **Code Editor:** A text editor where you can write, edit, and manage your Arduino code (sketches). It supports syntax highlighting to enhance code readability.
2. **Sketches and Libraries:** Arduino programs are referred to as "sketches." The IDE allows you to create, save, and organize sketches. It also includes a library manager for adding and managing external libraries to extend functionality.
3. **Compile and Upload:** The IDE provides buttons to compile the code into machine language (binary) and upload it to the Arduino board through a USB connection.
4. **Serial Monitor:** A tool within the IDE that allows you to communicate with the Arduino board by sending and receiving data through the serial port. This is useful for debugging and interacting with your Arduino project.
5. **Board Manager:** An interface to select the type of Arduino board you're working with, specifying the microcontroller type, clock frequency, and other settings.
6. **Examples:** A collection of pre-written example sketches that demonstrate various functionalities of the Arduino platform. These examples serve as valuable learning resources for beginners.

****

# Components Used

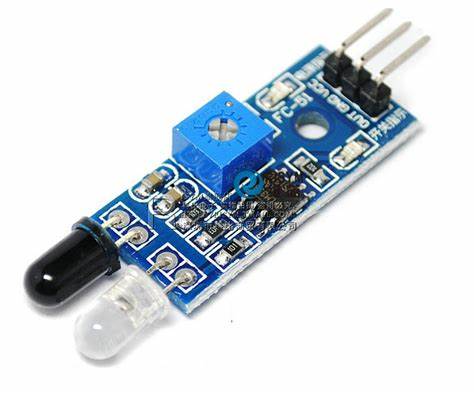
**Arduino UNO:-**

* It is a microcontroller board developed by Arduino.cc and is based on Atmega328 Microcontroller. The first Arduino project was started in Interaction Design Institute Ivrea in 2003 by David Cuartielles and Massimo Banzi with the intention of providing a cheap and flexible way for students and professionals to learn embedded programming.
* Arduino UNO is a very valuable addition in electronics that consists of a USB interface, 14 digital I/O pins(of which 6 Pins are used for PWM), 6 analog pins and an Atmega328 microcontroller. It also supports 3 communication protocols named Serial, I2C and SPI protocol. The software used for writing, compiling & uploading code to Arduino boards is called **Arduino IDE** (Integrated Development Environment),
* It has an operating voltage of 5V while the input voltage may vary from 7V to 12V. Arduino UNO has a maximum current rating of 40mA, so the load shouldn't exceed this current rating or you may harm the board.



**IR Proximity Sensor:-**

* An IR (Infrared) Proximity Sensor is a type of sensor that detects the presence or proximity of objects within its detection range without any physical contact. It works on the principle of emitting and receiving infrared light. When an object comes within the sensor's range, it reflects the emitted infrared light back to the sensor, which then detects this reflection and triggers a response.
* In Proximity Sensors an IR LED and a Photodiode is used to find an obstacle. The IR LED emits light in forwarding direction when an obstacle is ahead the light reflects and the Photodiode is activated. By this method, the obstacle is detected. These sensors are widely used in various applications such as automation, robotics, security systems, TV remote and Night-vision cameras and electronic devices. They are preferred for their non-contact operation, reliability, and cost-effectiveness

.

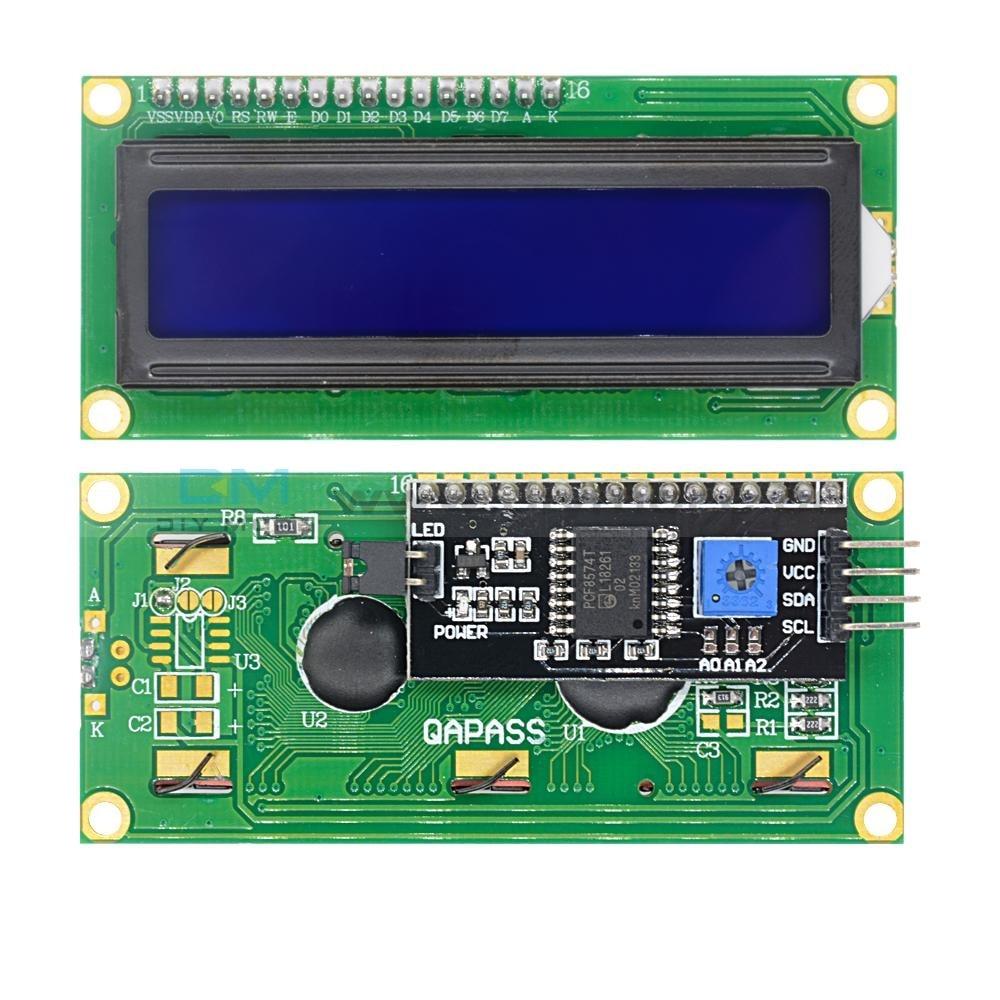
**Servo Motor:-**

* A servo motor is a rotary actuator that allows for precise control of angular position, velocity, and acceleration. It consists of a motor coupled with a sensor for position feedback and a control circuit. Servo motors are commonly used in various applications that require accurate and controlled rotational motion.
* The motor then rotates or moves according to the control signal and changes its position, speed, or torque, and sends a new feedback signal to the controller.The process repeats until the error signal becomes zero or negligible, indicating that the output shaft has reached the desired set point.



**16x2 LCD i2c Display:-**

The 16x2 LCD I2C display is a liquid crystal display (LCD) module with a dimension of 16 characters in each of the two rows, making a total of 32 characters. It is commonly used for displaying alphanumeric characters and simple graphics in embedded systems and electronic projects. What sets it apart from traditional 16x2 LCD displays is its integration with an I2C (Inter-Integrated Circuit) interface, which simplifies the wiring and allows for easier communication with microcontrollers.



**FLOWCHART**

Include Libraries

Initialize pins

Setup Function

Setup Loop

Check IR1 Status

IR1 is LOW?

**NO**

Check IR2 Status

IR2 is LOW?

**NO**

Welcome Screen

Update Slot Count

Infinite Loop

**ALGORITHM**

**1. Initialize libraries and variables:**

* Include necessary libraries: Wire.h, LiquidCrystal\_I2C.h, Servo.h
* Initialize LCD object (0x27 address, 16x2 dimensions)
* Initialize Servo object
* Define pin numbers for IR sensors, LED, and parking slot count
* Initialize flags for IR sensor detection (flag1 and flag2)
* Set initial number of parking slots

**2. Setup function:**

* Begin serial communication (9600 baud rate)
* Initialize LCD and backlight
* Set pin modes for IR sensors, LED, and Servo
* Attach Servo to pin 4
* Set initial Servo position (angle = 100)
* Display welcome message on LCD
* Wait for 2 seconds and clear the LCD

**3. Main loop:**

* Check if IR1 sensor detects a vehicle and flag1 is not set:
* If there are available parking slots:
* Set flag1
* If flag2 is not set, move the Servo to open the gate and decrement the slot count
* Otherwise, if parking is full:
* Display "SORRY :(" and "Parking Full" messages on LCD
* Turn on the LED for 5 seconds
* Clear the LCD and turn off the LED
  1. **Check if IR2 sensor detects a vehicle and flag2 is not set:**
* Set flag2
* If flag1 is not set and there are available slots:
* Move the Servo to open the gate and increment the slot count
* If both flags are set (vehicles detected at both entrances):
* Delay for 1 second
* Move the Servo to close the gate
* Reset flag1 and flag2
* Update LCD display with welcome message and current slot count

**5. Repeat the main loop indefinitely.**

# PROGRAM

* **Arduino 16x2 LCD I2C Scanner Code:-**

// Code to find LCD Address

/\*Analog Pin 4 – SDA

Analog pin 5 – SCL

Vcc – 5V

GND - GND

\*/

#include <Wire.h>

void setup()

{

Wire.begin();

Serial.begin(9600);

Serial.println("\nI2C Scanner");

}

void loop()

{

byte error, address;

int Devices;

Serial.println("Scanning...");

Devices = 0;

for(address = 1; address < 127; address++ )

{

Wire.beginTransmission(address);

error = Wire.endTransmission();

if (error == 0)

{

Serial.print("I2C device found at address 0x");

if (address<16)

Serial.print("0");

Serial.print(address,HEX);

Serial.println(" !");

Devices++;

}

else if (error==4)

{

Serial.print("Unknown error at address 0x");

if (address<16)

Serial.print("0");

Serial.println(address,HEX);

}

}

if (Devices == 0)

Serial.println("No I2C devices found\n");

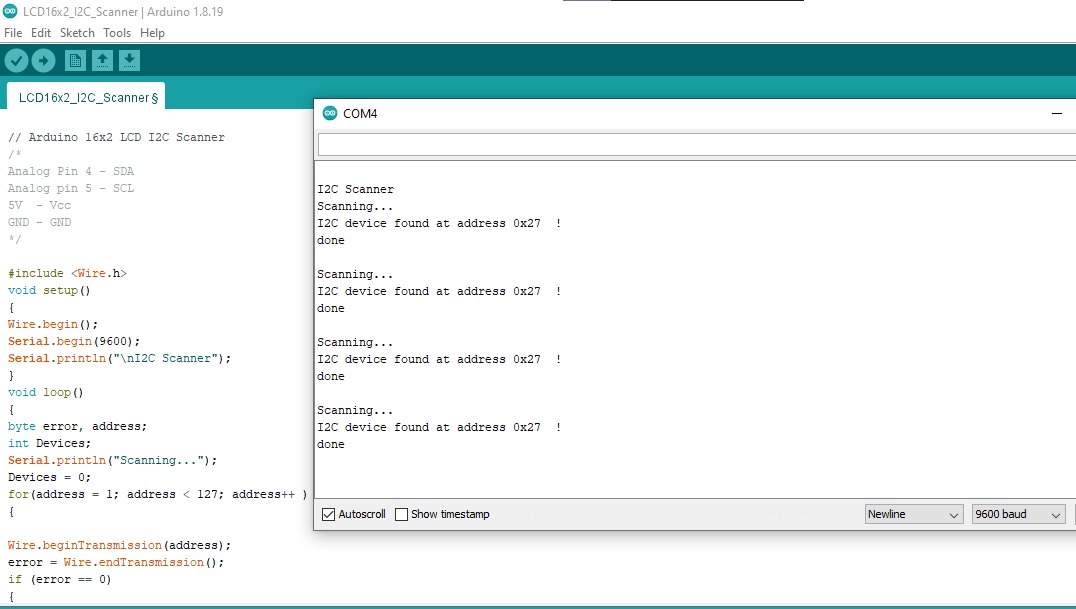
else

Serial.println("done\n");

delay(5000);

}

**Output(Serial Monitor):-**



* **Main Code:-**

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

LiquidCrystal\_I2C lcd(0x27,16,2);

#include <Servo.h>

Servo myservo;

int IR1 = 2;

int IR2 = 3;

int Slot = 4; // Total number of parking Slots

int flag1 = 0;

int flag2 = 0;

const int LED\_PIN = 8; // Define the pin for the LED

void setup() {

Serial.begin(9600);

lcd.init(); // initialize the lcd

lcd.backlight(); // open the backlight

pinMode(IR1, INPUT);

pinMode(IR2, INPUT);

pinMode(LED\_PIN, OUTPUT); // Initialize the LED pin

myservo.attach(4);

myservo.write(100);

lcd.setCursor (0,0);

lcd.print(" Automated ");

lcd.setCursor (0,1);

lcd.print(" PARKING SYSTEM ");

delay (2000);

lcd.clear();

}

void loop(){

if(digitalRead (IR1) == LOW && flag1==0){

if(Slot>0){

flag1=1;

if(flag2==0){

myservo.write(0);

Slot = Slot-1;

}

} else {

lcd.setCursor (0,0);

lcd.print(" SORRY :( ");

lcd.setCursor (0,1);

lcd.print(" Parking Full ");

digitalWrite(LED\_PIN, HIGH); // Turn on the LED

delay (5000);

lcd.clear();

digitalWrite(LED\_PIN, LOW); // Turn off the LED when parking is not full

}

}

if(digitalRead (IR2) == LOW && flag2==0){

flag2=1;

if(flag1==0 && Slot < 4){ // Ensure there are available slots before increasing the count

myservo.write(0);

Slot = Slot+1;

}

}

if(flag1==1 && flag2==1){

delay (1000);

myservo.write(100);

flag1=0;

flag2=0;

}

lcd.setCursor (0,0);

lcd.print(" WELCOME! ");

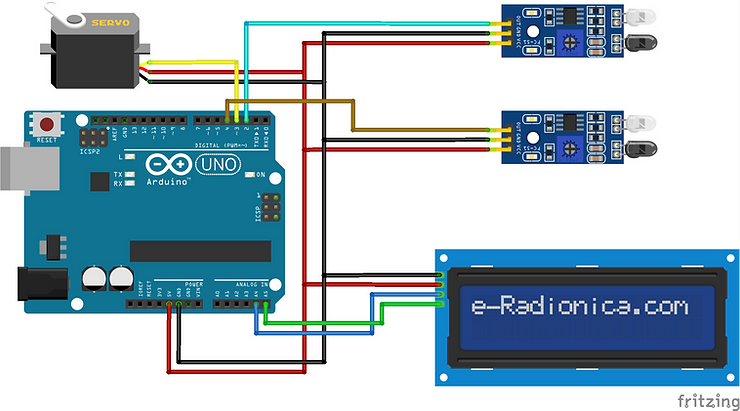
lcd.setCursor (0,1);

lcd.print("Slot Left: ");

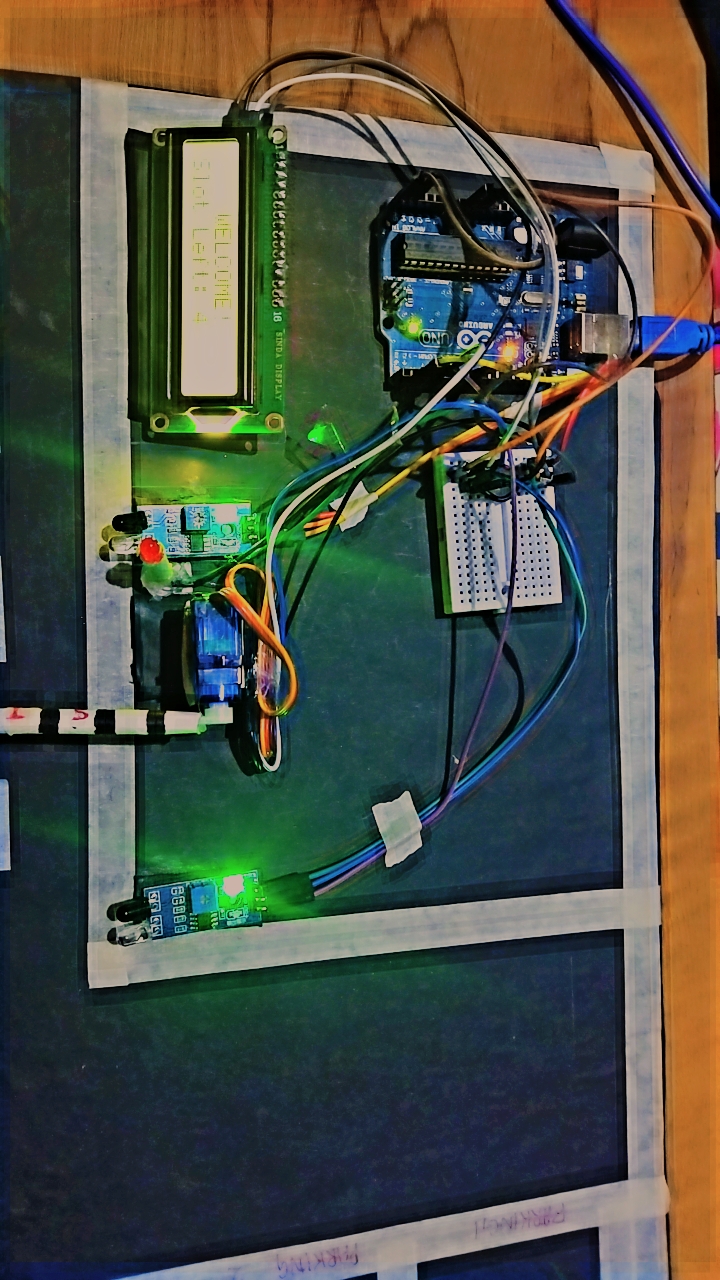
lcd.print(Slot);

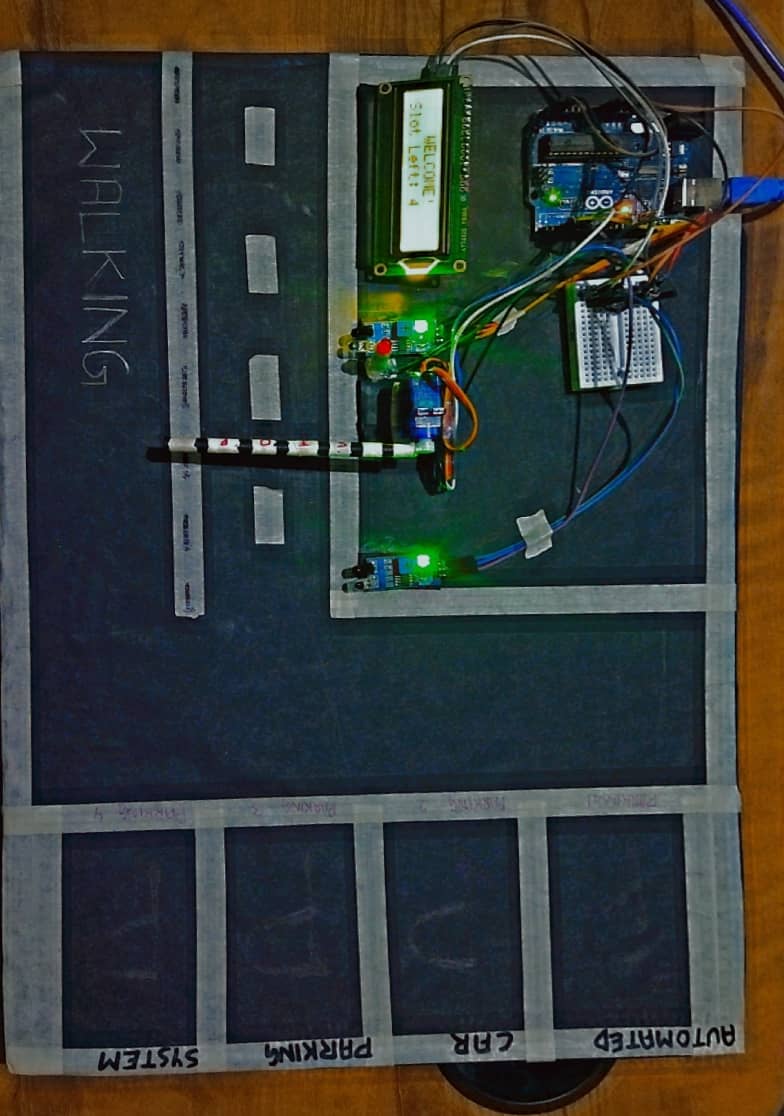
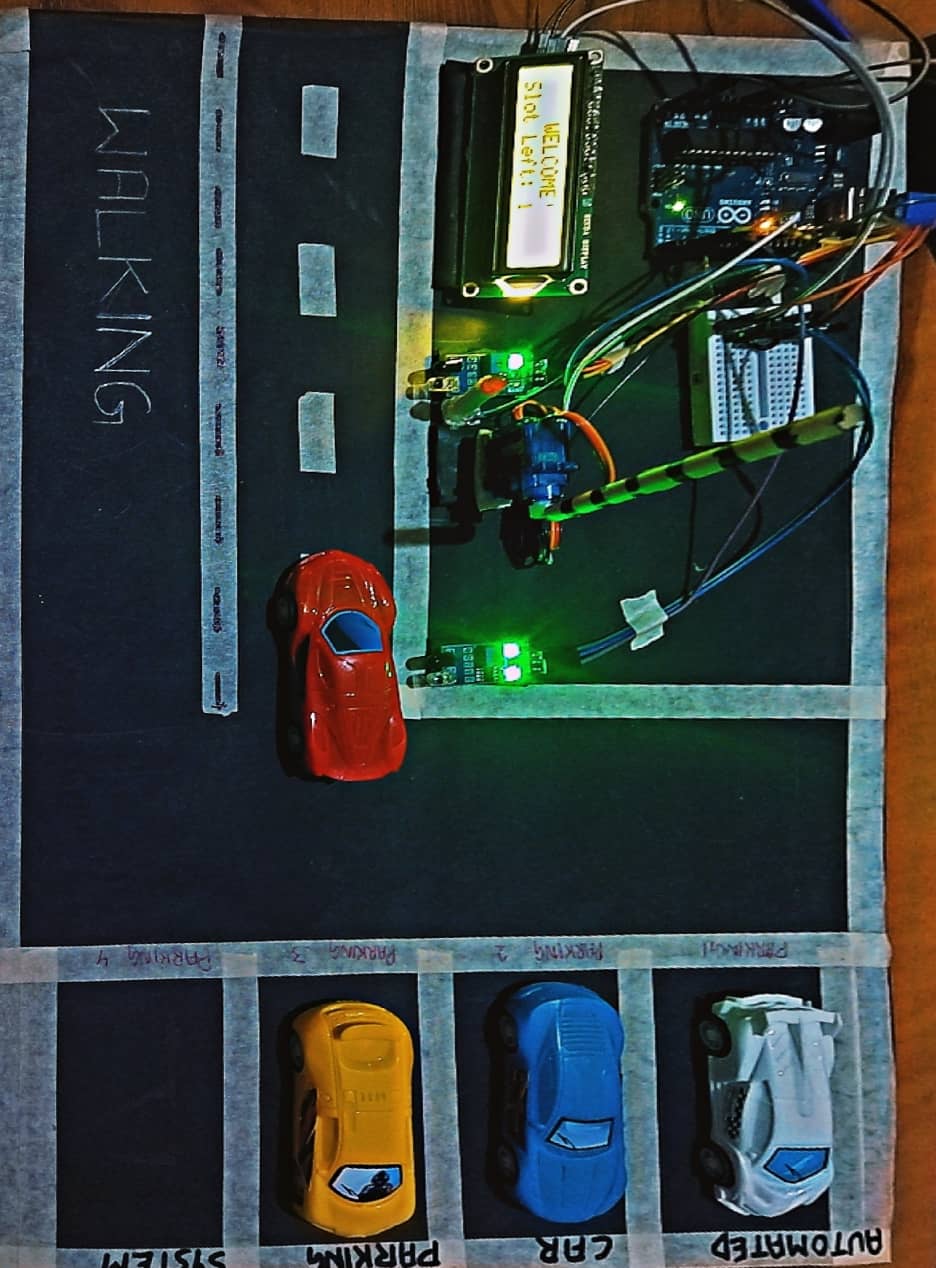
}

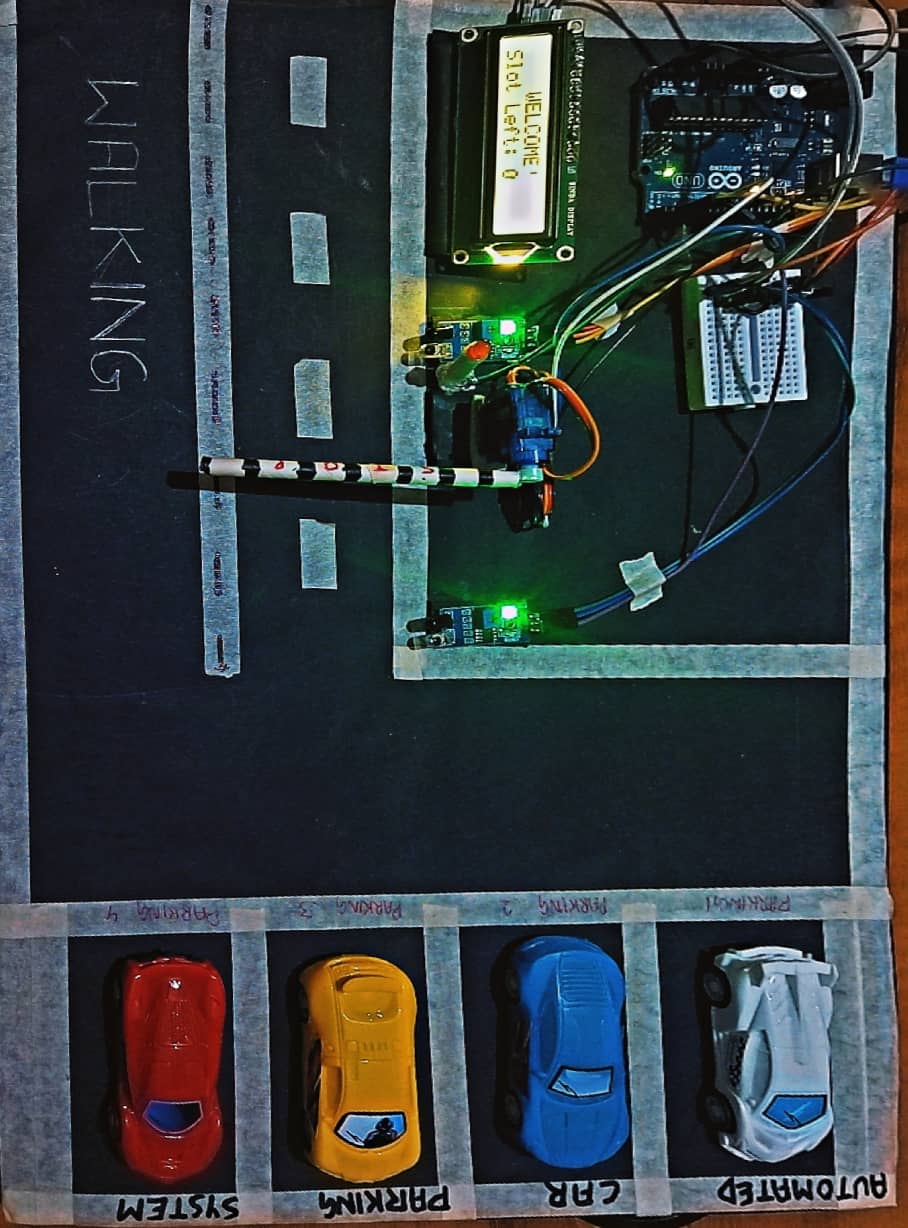
**CIRCIUT DIAGRAM**

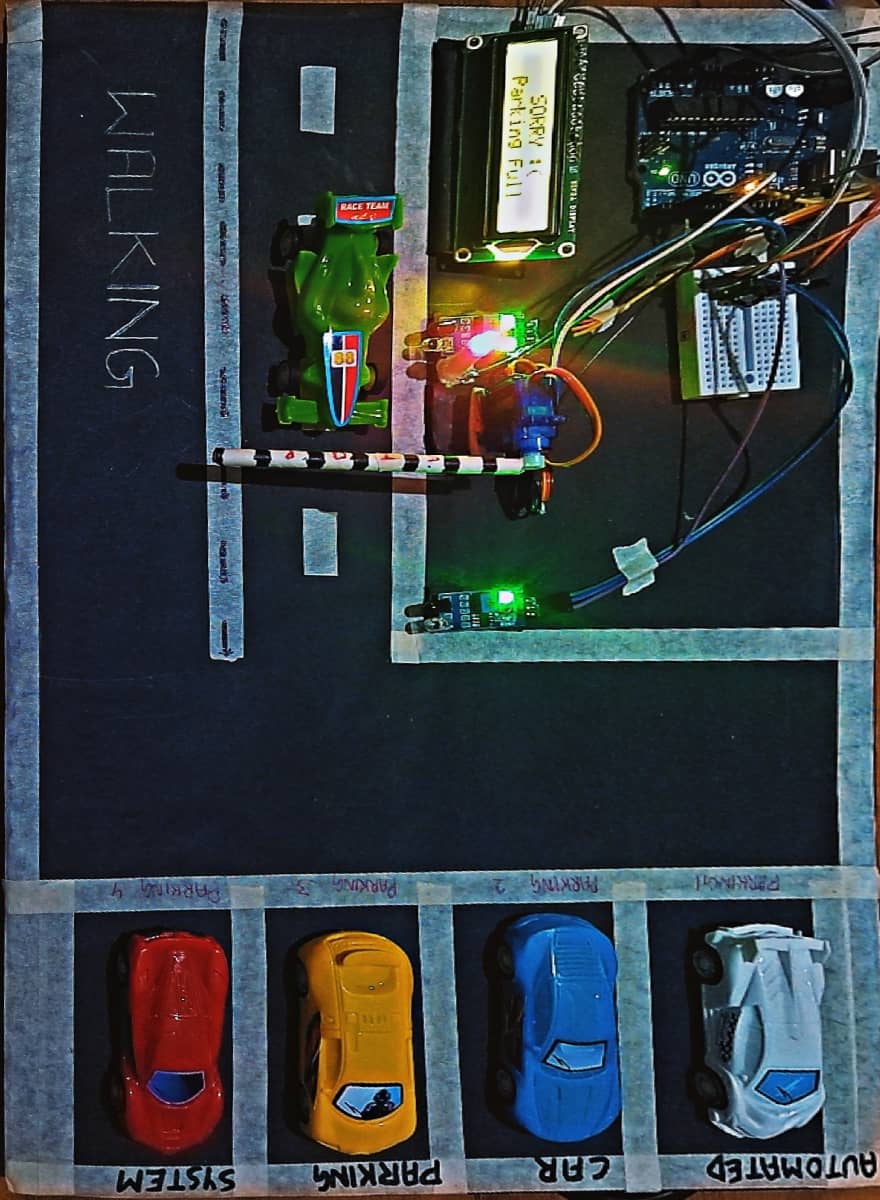


**Hardware:-**

****

**RESULT**  ****

****

****

# CONCLUSION

* In conclusion,The Automated Parking System project utilizing Arduino Uno is a testament to the potential of microcontrollers in creating smart and efficient solutions for everyday problems. By integrating IR sensors, a servo motor, LED Indicator and an LCD display, this system successfully manages parking spaces and automates gate control. This project not only enhances convenience but also optimizes parking utilization, demonstrating the practical applications of Arduino in real-world scenarios. The system can be further expanded by incorporating additional features such as automatic billing and mobile integration, making it an ideal solution for modern parking needs.
* Throughout the development and implementation of this project, key objectives were achieved. The integration of IR sensors with Arduino Uno enabled accurate vehicle detection, ensuring smooth operations within the parking facility. Servo motors controlled gate mechanisms, offering secure entry and exit for vehicles. LED indicators and LCD screens provided real-time status updates and user instructions, enhancing convenience and accessibility.

# 

# COURSE OUTCOME

The successful completion of this project has significantly contributed to the fulfilment of the following course outcomes.

* **CO-1:** Identify the IOT Components and its capabilities.
* **CO-6:** Design an IOT application with ML and Arduino/Raspberry Pi.

Hence through this PBL **CO1 & CO6** is Satisfied.

# References

# [Arduino Car Parking System (viralsciencecreativity.com)](https://www.viralsciencecreativity.com/post/arduino-car-parking-system)

# [Arduino Car Parking System (youtube.com)](https://www.youtube.com/watch?v=6gccSyp_uJQ&t=220s)

# [Arduino Smart Parking System (youtube.com)](https://www.youtube.com/watch?v=aMRMr4Iu2wY&t=330s)

# [Introduction to Arduino Uno - The Engineering Projects](https://www.theengineeringprojects.com/2018/06/introduction-to-arduino-uno.html)