Project Report

IOT BASED CAR PARKING SYSTEM USING HYDRAULIC LIFT

1.3 Abstract

This project presents an IoT-based car parking system employing hydraulic lifts to maximize space utilization and automate parking operations. The proposed system uses ESP32-based microcontrollers to control and monitor various components, including IR sensors for slot detection and servo motors for vehicle placement. Through real-time data collection and communication, users can access slot availability information and operate the parking system remotely via a dedicated interface. The inclusion of hydraulic lifts facilitates vertical parking, enabling the system to accommodate more vehicles within limited urban spaces. The project also incorporates scope for intelligent enhancements, such as time series forecasting for slot availability and car detection using convolutional neural networks (CNNs). By integrating IoT and automation, this system addresses critical urban parking challenges while paving the way for sustainable and scalable solutions.

1.4 SDG goal Alignment Justification

This project directly aligns with **SDG 11: Sustainable Cities and Communities**, which emphasizes making cities inclusive, safe, resilient, and sustainable. By integrating IoT technology and hydraulic lifts, the system addresses the growing challenge of urban parking inefficiencies and land scarcity. The vertical parking mechanism reduces the physical footprint required for parking lots, freeing up valuable urban space for green areas, pedestrian zones, or other community-enhancing developments. The automated and IoTenabled functionality minimizes congestion caused by vehicles searching for parking, leading to reduced fuel consumption and lower carbon emissions. This contributes to cleaner urban air quality and improved public health. The system's potential for integration with real-time data analytics and predictive features further supports the creation of smart city ecosystems, fostering sustainable transportation infrastructure and enhancing the overall quality of urban life. By addressing these critical aspects, the project embodies the principles of sustainability and resilience central to SDG 11.

2. Related Works

2.1 Literature Survey

1. **Md. Mejbaul Haque et al. (2022)** developed a reservation-based smart parking system that integrates IR sensors and Arduino for slot detection and management. It achieved high accuracy but faced challenges with sensor reliability and scalability
2. **Dhanabalraj et al. (2021)** presented a car parking allocation system using IR sensors and Arduino for real-time updates. Despite its cost-effectiveness, the scalability and accuracy of the system were limited due to basic sensors
3. **Iqbal et al. (2019)** proposed a GSM-integrated parking system using Arduino. It efficiently allocated parking slots but was constrained by single-point failures and dependency on GSM networks
4. **Patil et al. (2020)** created an IoT-based economic parking system using NodeMCU and RFID technology. It facilitated user-friendly operations but struggled with scalability and high implementation costs
5. **Lavanya et al. (2022)** introduced an IoT-driven system for parking detection using IR sensors and Arduino. The system was efficient but lacked features like online reservation and GPS integration
6. **Ma et al. (2017)** applied machine vision techniques in automatic parking systems, achieving improved parking efficiency. However, the system required enhancements for robustness and path-tracking accuracy
7. **Grbić & Koch (n.d.)** implemented YOLO-based parking slot detection with high robustness under various weather conditions. Camera angle sensitivity remained a notable limitation

8. **Elechi & Saturday (2022)** designed a barcode and ultrasonic sensor-based parking system. While it efficiently handled vertical parking, it lacked comprehensive path tracking details.
9. **Kabir et al. (2019)** presented a fee management-integrated parking system using Arduino and RFID. The system effectively automated entry and fee processes but relied heavily on manual intervention for recharging RFID tags
10. **Prabhakaran & Dhivya (2020)** used NodeMCU and RFID for a real-time IoT-based parking system with automated payment processing. Sensor reliability and network dependency were key challenges
11. **Archana et al. (2023)** integrated RFID, IR sensors, and Arduino for urban parking management. The system offered real-time updates but was limited by RFID vulnerabilities and environmental factors.
12. **Annirudh& Arun Kumar (2021)** proposed an IoT-based intelligent parking management system with mobile app integration. The initial setup cost and limited sensor coverage posed barriers
13. **Veeramanickam et al. (2022)** utilized ultrasonic sensors and cloud integration for a smart parking system. The FCFS scheduling method was effective but lacked user preference considerations
14. **Ratko Grbić & Brando Koch (2023)** focused on real-time empty slot detection using ultrasonic sensors. Environmental interference and maintenance costs were noted limitations.
15. **Kaushik (2023)** employed ML and advanced neural networks for cloud-integrated parking systems, achieving high prediction accuracy but with latency risks and high costs.
16. **Van-An VO et al. (2023)** applied CNNs for parking slot detection using image data, achieving scalability but requiring substantial computational resources and high-quality data.
17. **Reddy et al. (2023)** developed an RFID-based smart parking system using Arduino for efficient slot allocation and fee management. IR sensor limitations and maintenance needs were challenges.
18. **Yasmin et al. (2021)** implemented Mask R-CNN for deep learning-based parking slot detection, demonstrating high accuracy but requiring labor-intensive manual segmentation.
19. **Rajasekhar et al. (2021)** integrated IoT with motorized lifts for a multi-level parking system. Indoor applicability and network limits constrained its scalability.
20. **Nibedita Priyadarsini Mohapatra et al. (2024)** combined IoT and ML for vehicle detection, improving system intelligence but with high setup and maintenance costs

2.2 Comparative statement (Tabulation) and Research gap Summary

Paper	Methodology/Technology	Strengths	Limitations
Md. Mejbaul Haque et al. (2022)	IoT-based parking system with IR sensors and Arduino	High accuracy in reservation cycles	Limited scalability and sensor reliability
Dhanabalraj et al. (2021)	IR sensors and Arduino for real-time slot detection	Cost-effective, simple implementation	Limited to structured parking, sensor accuracy issues
Iqbal et al. (2019)	GSM and Arduino for slot allocation	Efficient slot allocation, SMS integration	Reliance on GSM network, singlepoint failures

Patil et al. (2020)	NodeMCU, RFID, and IoT integration	Real-time monitoring, userfriendly	High implementation cost, limited scalability
Lavanya et al. (2022)	IR sensors and Arduino for slot detection	Scalable design, reduces congestion	Lacks GPS and reservation features
Ma et al. (2017)	Machine vision and AdaBoost for parking scene recognition	Handles narrow spaces	Needs robustness improvements
Grbić & Koch (n.d.)	YOLO-based vehicle detection	High robustness and real-time updates	Camera angle sensitivity, calibration needs
Elechi & Saturday (2022)	Ultrasonic sensors and barcode systems	Efficient space usage	Limited to vertical parking
Kabir et al. (2019)	RFID and Arduino for fee management	Automated fee and slot allocation	Manual RFID recharges required
Prabhakaran & Dhivya (2020)	NodeMCU and IoT integration	Cost-effective, realtime updates	Network dependency, RFID tampering risks
Archana et al. (2023)	RFID and IoT for urban parking	Security through RFID, adaptability	Scalability limitations
Annirudh & Arun Kumar (2021)	IoT sensors and mobile app	Real-time updates, user-friendly	Initial high costs, limited coverage
Veeramanickam et al. (2022)	Ultrasonic sensors and cloud for FCFS scheduling	Efficient real-time allocation	Fixed allocation logic, energy consumption
Ratko Grbić & Koch (2023)	Ultrasonic sensor-based empty slot detection	Energy-efficient, analytics potential	High initial setup costs
Kaushik (2023)	ML and neural networks for predictions	High accuracy in forecasting	Latency risks, complexity
Van-An VO et al. (2023)	CNN for real-time slot detection	Scalable and accurate	High computational costs, privacy concerns
Reddy et al. (2023)	RFID and Arduino for smart parking	Enhanced security, low cost	Sensor limitations, maintenance needs
Yasmin et al. (2021)	Mask R-CNN for slot detection	High accuracy, robust under conditions	Manual segmentation required
Rajasekhar et al. (2021)	IoT multi-level system with ESP-NOW	Real-time slot updates, energyefficient	Limited to indoor setups

Mohapatra et al. (2024)	IoT and ML for parking detection	Enhanced accuracy and optimization	High setup and maintenance costs

2.3 Hardware Requirements

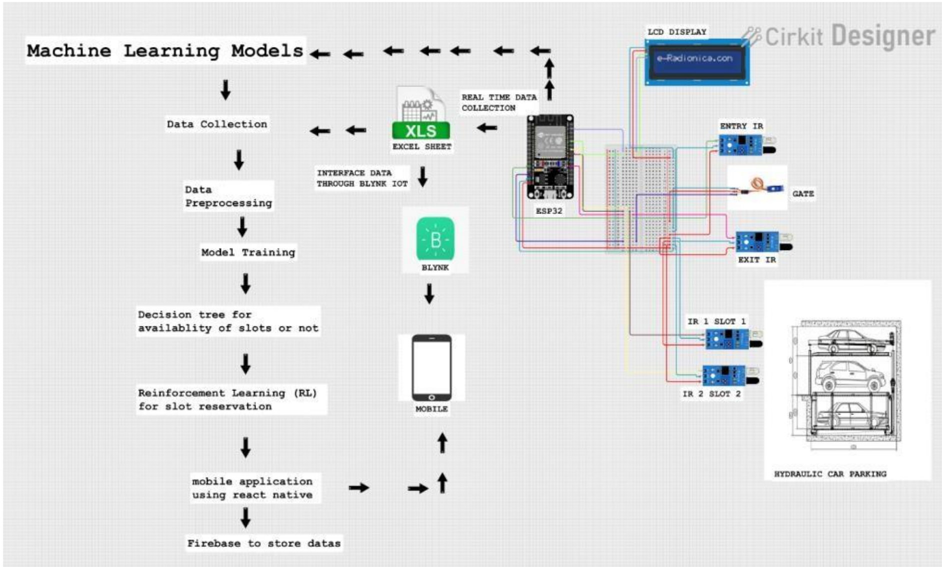
1. **Microcontroller:**
   - ESP32 microcontroller for IoT integration and system management.
2. **Sensors:**
   - Infrared (IR) sensors for slot detection.
3. **Actuators:**
   - Servo motors to operate entrance gates and manage hydraulic lifts.
4. **Hydraulic Lift:**
   - Mechanism for vertical parking with efficient space utilization.
5. **Communication Module:**
   - Wi-Fi module for IoT connectivity and data exchange.
6. **Display Units:**
   - LCD displays to show slot availability and other system statuses.
7. **Power Supply:**
   - Adequate power source to drive the microcontroller, sensors, and hydraulic components.

2.4 Software Requirements

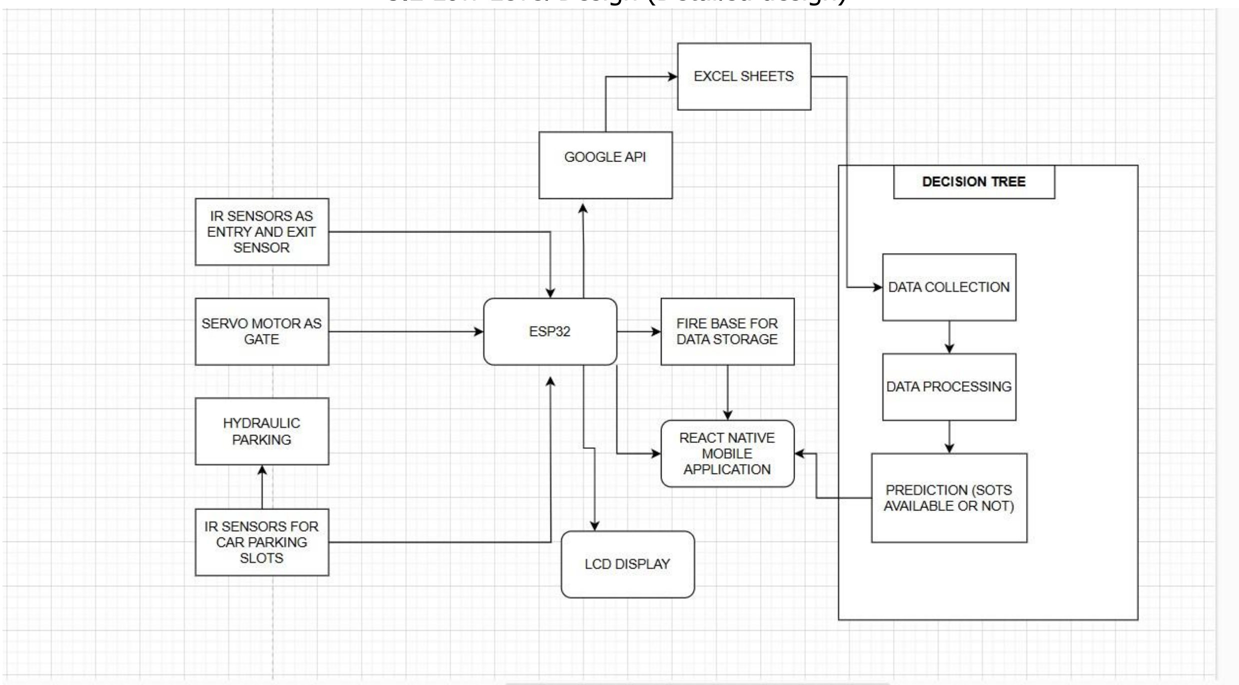
1. **Programming Tools:**
   - Arduino IDE for programming the ESP32 microcontroller.
   - Python or C++ for additional system logic.
2. **IoT Platforms:**
   - Blynk for real-time monitoring and data visualization.
3. **Mobile Application Development:**
   - React Native for creating user-friendly apps for slot reservation and control.
4. **Communication Protocols:**
   - HTTP protocols for data exchange between devices and servers.
5. **Cloud Integration:**
   - Firebase IoT Core for storing and processing real-time data.
6. **Control Algorithms:**
   - slot allocation.

3. System Design

3.1 High-Level Design



3.2 Low-Level Design (Detailed design)



3.3 Methodology

The methodology for the IoT-based car parking system using hydraulic lifts involves two key aspects: hardware implementation and software integration. The hardware includes IR sensors for detecting slot availability, an ESP32 microcontroller for processing data, servo motors for operating the hydraulic lifts, and a power supply unit with rechargeable batteries for stable operation. When a vehicle arrives, IR sensors detect slot availability, and the ESP32 processes this information, sending commands to the hydraulic lift to move the vehicle to the designated level. The software side utilizes the Blynk IoT platform for real-time monitoring and control, allowing users to view slot availability, reserve slots, and track their vehicles via a smartphone app or web dashboard. All data, including slot status, vehicle entry/exit times, and lift operations, is logged in a cloud database for analysis and reporting. This integrated system streamlines parking operations, improves space utilization, and minimizes manual intervention.

4. Results and Discussion

4.1 Implementation Code and Results

```
 #include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <ESP32Servo.h>
#include <WiFi.h>
#include <FirebaseESP32.h>  // Firebase ESP32 library #include
<HTTPClient.h>
```

```
// Create an LCD object with I2C address 0x27, 16 columns, and 4 rows
LiquidCrystal_I2C lcd(0x27, 16, 4);

// Create a Servo object
Servo myservo;

// Define pin connections for sensors and servos
#define ir_enter 14
#define ir_back  4
#define ir_car1 5   // First car slot sensor
#define ir_car2 18  // Second car slot sensor
#define ir_car3 19  // Third car slot sensor
#define servo_pin 13  // Define the servo pin, adjust as per your connection

int S1 = 0, S2 = 0, S3 = 0; int flag1 = 0, flag2 = 0; int slot = 3;  // Total
number of slots is now 3 const char* ssid = "Magesh's Pixel 8"; const
char* password = "12345677"; const char* firebaseDatabaseURL =
"https://parking-8f5fc-default-rtdb.asiasoutheast1.firebasedatabase.app/";
// Update with your Firebase URL

void setup() {
  Serial.begin(115200);  // Use 115200 baud rate for ESP32

  // Set up pin modes for sensors
pinMode(ir_car1, INPUT_PULLUP);
pinMode(ir_car2, INPUT_PULLUP);
pinMode(ir_car3, INPUT_PULLUP);
pinMode(ir_enter, INPUT_PULLUP);
  pinMode(ir_back, INPUT_PULLUP);

  // Attach servo to its pin
myservo.attach(servo_pin);
  myservo.write(90);  // Initial position of the servo

  // Initialize the LCD with 16 columns and 4 rows
  lcd.begin();
lcd.setCursor(0, 1);
lcd.print(" Car Parking ");
lcd.setCursor(0, 2);
lcd.print("   System    ");
delay(2000);
  lcd.clear();

  // Read initial sensor values
Read_Sensor();
updateSlotAvailability();
connectToWiFi();
}

void loop() {
```

```
 Read_Sensor();
updateSlotAvailability();

  // Display available slots on the first
line   lcd.setCursor(0, 0);
lcd.print("Have Slot: ");
  lcd.print(slot);
lcd.print("   ");

  // Display the status of each slot on the second line and scroll it
  String slotStatus = "S1:" + String(S1 ? "Fill " : "Empty ") + "S2:" + String(S2 ? "Fill " : "Empty ") +
"S3:" + String(S3 ? "Fill " : "Empty ");
lcd.setCursor(0, 1);
  lcd.print(slotStatus);

  // Scroll the status message   for (int i = 0; i <
slotStatus.length() - 15; i++) {
lcd.scrollDisplayLeft();  // Scroll the text to the left
delay(300);  // Delay for smooth scrolling
  }

  // Entry gate logic
  if (digitalRead(ir_enter) == LOW && flag1 == 0) {     if (slot > 0)
{  // Only allow entry if there are slots available       flag1 = 1;
if (flag2 == 0) {         myservo.write(180);  // Open the gate
slot = slot - 1;  // Decrease available slots         sendToFirebase(S1
? "ir1" : S2 ? "ir2" : "ir3", String(slot), slot);       }     } else {
lcd.setCursor(0, 0);       lcd.print(" Parking Full ");
      delay(1500);  // Display the "Parking Full" message
    }
  }

  // Exit gate logic   if (digitalRead(ir_back) ==
LOW && flag2 == 0) {     flag2 = 1;     if (flag1
== 0) {       myservo.write(180);  // Open the
gate       slot = slot + 1;  // Increase available
slots
      sendToFirebase("Free", String(slot), slot);
    }
  }

  // Reset flags and close gate
if (flag1 == 1 && flag2 == 1) {
    delay(1000);  // Delay to simulate gate operation
myservo.write(90);  // Close the gate
    flag1 = 0;
flag2 = 0; }

 delay(100);  // Short delay to avoid rapid looping
```

```
}

// Function to read sensor states
void Read_Sensor() {
  // Update sensor values based on readings
  S1 = (digitalRead(ir_car1) == LOW) ? 1 : 0;
  S2 = (digitalRead(ir_car2) == LOW) ? 1 : 0;
  S3 = (digitalRead(ir_car3) == LOW) ? 1 : 0;
}

// Update slot availability and send data to
Firebase void updateSlotAvailability() {   if (S1 &&
S2 && S3) {
    slot = 0;     sendToFirebase("All Full",
String(slot), slot);
  } else if (!S1 && !S2 && !S3) {
    slot = 3;
    sendToFirebase("Free", String(slot), slot);
  } else {     slot = 3 - (S1 + S2 + S3);  // Update the available
slot count     sendToFirebase(S1 ? "ir1" : S2 ? "ir2" : "ir3",
String(slot), slot);   }
}

// Connect to WiFi void connectToWiFi() {
WiFi.begin(ssid, password);   while
(WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("Connected to WiFi");
}

// Send data to Firebase Realtime Database void
sendToFirebase(String slot, String status, int availableSlots) {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;

    // URL for the specific slot node
    String url = String(firebaseDatabaseURL) + "/slots/" + slot + ".json";
http.begin(url);
    http.addHeader("Content-Type", "application/json");

    // JSON payload for the slot status
    String jsonPayload = "{\"status\":\"" + status + "\",\"sensor\":\"" + slot + "\"}";
int httpResponseCode = http.PATCH(jsonPayload);

    // Check response     if
(httpResponseCode > 0) {
```

```
Serial.print("Slot update HTTP
Response code: ");
     Serial.println(httpResponseCode);
   } else {
      Serial.print("Error on sending PATCH for slot: ");
      Serial.println(httpResponseCode);
    }
    http.end();

    // Update the availability node
    http.begin(String(firebaseDatabaseURL) + "/slots/availability.json");     http.addHeader("Content-
Type", "application/json");
    jsonPayload = "{\"availableSlots\":" + String(availableSlots) + "}";
httpResponseCode = http.PATCH(jsonPayload);

    if (httpResponseCode > 0) {
      Serial.print("Availability update HTTP Response code: ");
      Serial.println(httpResponseCode);
    } else {
      Serial.print("Error on sending PATCH for availability: ");
      Serial.println(httpResponseCode);
    }
    http.end();
  } else {
    Serial.println("Error in WiFi connection");
  }
}
```

4.2 Metrics

1. Parking Slot Metrics

- Occupancy Status: Tracks if a slot is occupied or vacant.
- Total Slots Available: Number of free parking spaces.
- Reservation Status: Indicates if a slot is reserved or not.

2. Sensor Data Metrics

- Infrared Sensor Readings: Detects car presence (e.g., ir1 and ir2 for slots).

3. User Interaction Metrics

- Reservation Count: Number of slots reserved by users.
- Average Reservation Duration: Tracks typical reservation periods (e.g., 1, 2, or 3 hours).
- Payment Metrics: Monitors payment statuses and payment methods (e.g., UPI, bank transfer).
- Entry/Exit Rates: Tracks how often vehicles enter and leave the parking area.

4. Financial Metrics

- Revenue Generated: Based on parking fees collected (normal or reserved).
- Payment Success Rate: Percentage of successful transactions.
- Fee Collection Breakdown: Revenue by payment type (UPI, bank, etc.).

5. System Performance Metrics

- Latency in Data Updates: Time taken for parking data to sync with the database (Firebase or others).
- Error Rates: Counts sensor errors, reservation conflicts, or payment issues.
- Device Uptime: Tracks the operational status of ESP32, sensors, and LCD.

4.3 Results in table/Graph/Data(No screenshots, only text form of data in table), Graph should be drawn using Excel or python

Entry date and Time	Car parking slot	Availability
2024-09-15 17:00:32	Free	2
2024-09-15 17:00:35	Free	2
2024-09-15 17:00:37	Free	2
2024-09-15 17:00:40	Free	2
2024-09-15 17:00:43	Free	2
2024-09-15 17:00:45	Free	2
2024-09-15 17:00:48	Free	2
2024-09-15 17:00:51	Free	2
2024-09-15 17:00:53	Free	2
2024-09-15 17:00:56	Free	2
2024-09-15 17:00:58	Free	2
2024-09-15 17:01:01	Free	2
2024-09-15 17:01:03	Free	2
2024-09-15 17:01:06	Free	2
2024-09-15 17:01:08	Free	2
2024-09-15 17:01:10	ir2	1
2024-09-15 17:01:13	ir1	1
2024-09-15 17:01:15	ir1	1
2024-09-15 17:01:17	ir1	1

2024-09-15 17:01:20	ir1	1
2024-09-15 17:01:22	ir1	1
2024-09-15 17:01:25	Both Full	0
2024-09-15 17:01:27	Both Full	0
2024-09-15 17:01:30	Both Full	0
2024-09-15 17:01:33	ir2	1
2024-09-15 17:01:35	ir2	1
2024-09-15 17:01:38	ir2	1
2024-09-15 17:01:40	ir2	1
2024-09-15 17:01:43	ir2	1
2024-09-15 17:01:45	ir2	1
2024-09-15 17:01:48	ir2	1
2024-09-15 17:01:50	ir2	1
2024-09-15 17:01:53	Free	2
2024-09-15 17:01:55	Free	2
2024-09-15 17:01:58	Free	2
2024-09-15 17:02:01	Free	2
2024-09-15 17:02:04	Free	2
2024-09-15 17:02:08	Both Full	0
2024-09-15 17:02:11	Both Full	0
2024-09-15 17:02:14	Both Full	0
2024-09-15 17:02:17	Both Full	0
2024-09-15 17:02:20	Both Full	0
2024-09-15 17:02:23	Both Full	0

2024-09-15 17:02:25	Both Full	0
2024-09-15 17:02:28	Free	2
2024-09-15 17:02:30	Free	2
2024-09-15 17:02:33	Free	2
2024-09-15 17:02:35	Free	2
2024-09-15 17:02:38	Free	2
2024-09-15 17:02:40	Free	2
2024-09-15 17:02:42	Free	2
2024-09-15 17:02:45	Free	2
2024-09-15 17:02:48	Free	2
2024-09-15 17:02:50	Free	2
2024-09-15 17:02:53	Free	2
2024-09-15 17:02:55	Free	2
2024-09-15 17:02:58	Free	2
2024-09-15 17:03:00	Free	2
2024-09-15 17:03:03	Free	2
2024-09-15 17:03:08	Free	2
2024-09-15 17:03:11	Free	2
2024-09-15 17:03:13	Free	2
2024-09-15 17:03:16	Free	2
2024-09-15 17:03:18	Free	2
2024-09-15 17:03:21	Free	2
2024-09-15 17:03:24	Free	2

2024-09-15 17:03:26	Free	2
2024-09-15 17:03:29	Free	2
2024-09-15 17:03:31	Free	2
2024-09-15 17:03:34	Free	2
2024-09-15 17:03:37	Free	2
2024-09-15 17:03:39	Free	2
2024-09-15 17:03:41	Free	2
2024-09-15 17:03:44	Free	2
2024-09-15 17:03:46	Free	2
2024-09-15 17:03:49	Free	2
2024-09-15 17:03:51	Free	2
2024-09-15 17:03:53	Free	2
2024-09-15 17:03:55	Free	2
2024-09-15 17:03:58	Free	2
2024-09-15 17:04:01	Free	2
2024-09-15 17:04:03	Free	2
2024-09-15 17:04:05	Free	2
2024-09-15 17:04:08	Free	2
2024-09-15 17:04:10	Free	2
2024-09-15 17:04:13	Free	2
2024-09-15 17:04:15	Free	2
2024-09-15 17:04:18	Free	2
2024-09-15 17:04:20	Free	2
2024-09-15 17:04:23	Free	2

2024-09-15 17:04:25	Free	2
2024-09-15 17:04:27	ir2	1

4.4 Mapping the results with problem statement an
systems

Mapping results with problem statement:
In our problem statement, we aimed to develop an
vehicle movement. Our implemented solution achie
slots in real-

minimizing manual intervention, and reducing time
retrieval.

Mapping results with existing systems:

multi-level parking. This approach imp
Furthermore, existing systems often lack a dynami
vehicle movement, which is a core feature of c

4.5 Discussions

challenge data due to environmental factors was

managed through

requirements of an automated car parking solution

5. Conclusion and Future Developments

Our IoT-

hydraulic lifts, the system addresses the limita
populated urban areas.
Future Developments:

1.
   based on historical and real-time data.
2. Image Recognition Integration: Adding car
   with a CNN plates for automated vehicle
   identification and slot assignment.
3. Mobile Application: Developing a user-
   availability, and track their vehicle's status.
4.
   sources to improve sustainability.

-based car parking system utilizing hydraulic lifts
ESP32 microcontrollers, IR sensors, and servo motors
to

me and controlling the hydraulic lift system to park
or
ur goal by ensuring a streamlined parking process,

king allocations or require manual validation of slot
ates slot detection but also integrates hydraulic lifts for
ban areas.
-time feedback loop for managing slot availability

s, ESP32 modules, and servo motors is critical for the
nsors is essential to ensure accurate slot detection. One
able power supply to the hydraulic lift and
r optimal performance. Additionally, noise in the sensor
y. Despite these challenges, the system effectively

draulic lifts has demonstrated the capability to manage
parking
automating the parking and retrieval process with tions
nventional parking systems, especially in densely

orating predictive algorithms to forecast slot availability

-based system to recognize license

iendly app to allow users to reserve slots, monitor

e system using solar panels or other renewable energy

6.Student Feedback (Student Experience in this Course Project)

This project provided valuable insights into designing an IoT system involving multiple hardware components like IR sensors, ESP32 microcontrollers, and servo motors. I gained hands-on experience in programming the ESP32 modules and integrating them with the hydraulic lift mechanism.
Initially, we faced challenges in synchronizing sensor inputs with servo motor actions and managing power supply for consistent operation. Additionally, debugging fluctuations in sensor readings required the implementation of robust data filtering techniques. Overall, this project enhanced my understanding of IoT systems, hardware-software integration, and problem-solving in real-world scenarios.

7. References

1. Haque MM, Rahman MA, Gani MM. A reservation-based smart parking system for urban areas. In 2022 International Conference on Electrical and Computer Engineering (ICECE) 2022 Dec 1 (pp. 15). IEEE.
2. Dhanabalraj P, Sherin JJ, Gopinath L, Kumar K, Gowthaman GM. Car parking allocation system using Arduino. In 2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS) 2021 Feb (pp. 1223-6). IEEE.
3. Iqbal BJ, Catherine KB, Mohan CP, Joseph CM, Sajitha PS. Arduino-based quest for parking slot using GSM and IR sensors. International Journal of Innovative Research in Science, Engineering and Technology. 2019 Jun;8(6):7347-51.
4. Patil M, Chakole V, Chetepawad K. IoT-based economic smart vehicle parking system. In Third International Conference on Intelligent Sustainable Systems (ICISS) 2020 Feb (pp. 1337-40). IEEE.
5. Lavanya R, Jyothsna R, HemaSri V, Lakshmi MMS, Apsana S, Mounika P, Pavani SL. IoT-based vehicle parking place detection. Proceedings of International Conference on Computing and Communication Technologies. 2022 Jun.
6. Ma S, Jiang H, Han M, Xie J, Li C. Research on automatic parking systems based on parking scene recognition. IEEE Access. 2017;5:1-11.
7. Grbić R, Koch B. Automatic vision-based parking slot detection and occupancy classification. Faculty of Electrical Engineering, Computer Science, and Information Technology Osijek.
8. Elechi P, Saturday N. Improved automated car parking system. Microelectronic Engineering. 2022 Oct.
9. Kabir AZMT, Nath ND, Hasan F, Utshaw RA, Saha L. Automated parking system with fee management using Arduino. In 10th ICCCNT 2019 July 6-8 (pp. 1-5). IEEE.
10. Prabhakaran K, Dhivya R. RFID-based automatic car parking system using IoT. In 2020 International Conference on Emerging Trends in Engineering (ICETE) 2020 Jan (pp. 1-4). IEEE.
11. Archana M, Raju SS, Preethi D, Mydhili SK, Vinothkumar U. Smart automated parking system using IoT. In International Conference on Sustainable Computing and Data Communication Systems (ICSCDS) 2023 Aug (pp. 1579-81). IEEE.
12. Annirudh D, Arun Kumar D. IoT-based intelligent parking management system. In 2021 IEEE Second International Conference on Control, Measurement and Instrumentation (CMI) 2021 (pp. 1-5). IEEE.
13. Veeramanickam MRM, Venkatesh B, Bewoor LA, Bhowte YW, Moholkar K, Bangare JL. IoT-based smart parking model using Arduino UNO with FCFS priority scheduling. ScienceDirect. 2022.
14. Grbić R, Koch B. Automatic car parking with empty slot detection. Expert Systems with Applications. 2023 Sep.
15. Kaushik P. Enhanced cloud car parking system using ML and advanced neural network. International Journal of Research in Science and Technology. 2023 Jan-Mar;13(1):1-10.
16. VO VA, PHAN VD, BUI VM, DO TN. Design of deep learning model applied for smart parking system. ResearchGate. 2023 Dec.

17. Reddy KVV, Zaki D, Shashikanth B, Tharun B, Harshini G. A novel IoT-based smart parking solution. In 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT) 2023 Jul (pp. 1-5). IEEE.
18. Yasmin T, Ahmed A, Irfan M, Ameer U, Alhumyani H. Automated vehicle parking slot detection system using deep learning. In 2021 International Conference on Artificial Intelligence (ICAI) 2021 Mar (pp. 54-9). IEEE.
19. Rajasekhar N, Maya P, Panday S. IoT-enabled multi-level smart parking system. In 2021 IEEE Second International Conference on Electronics and Sustainable Communication Systems (ICESC) 2021 Jun (pp. 774-80). IEEE.
20. Mohapatra NP, Pattanaik SR, Mohapatra A, Mishra SK. Smart automated parking system employing Internet of Things and machine learning methods to detect vehicles. In International Conference on Emerging Systems and Intelligent Computing (ESIC) 2024 (pp. 112-9). IEEE.