Project Report

on

Car Parking System

Submitted to

LOVELY PROFESSIONAL UNIVERSITY

in partial fulfilment of the requirements for the award of degree of

Master of Computer Applications

Submitted By:

Aryan Sharma (12220735)

Alok kr. Gupta (12220300)

Ashis Pradhan (12220083)

PSV Ganesh (12220402)

Supervised By:

Mohammad Asif Ikbal (22797)

LOVELY FACULTY OF TECHNOLOGY & SCIENCES

LOVELY PROFESSIONAL UNIVERSITY

PUNJAB

04/24

Table of Content

INTRODUCTION

In urban environments where space is limited and the number of vehicles continues to rise, efficient management of parking spaces has become a pressing issue. Traditional manual methods of parking management often lead to inefficiencies, frustration among users, and congestion in parking facilities. To address these challenges, automated car parking systems have emerged as innovative solutions to optimize parking space utilization and enhance user experience. A car parking system is an automated solution designed to manage and optimize the use of parking spaces in various facilities, including commercial complexes, residential buildings, and industrial sites. It employs advanced technologies such as sensors, microprocessors, and infrared communication to monitor the availability of parking slots in real-time. These devices are installed in each parking space to detect the presence of vehicles. They relay information to a central control unit, enabling real-time monitoring of parking space occupancy. The central control unit processes data received from sensors and manages the overall operation of the parking system. It calculates available parking spaces, updates status displays, and facilitates communication with users. Communication Interface: An interface, typically featuring displays at the entrance and throughout the parking facility, provides users with information about available parking spots. This interface may include LED displays, mobile applications, or web-based platforms. Upon entering the parking facility, users are greeted with real-time information indicating the availability of parking spots. The system directs them to vacant spaces, eliminating the need for extensive manual searching. As vehicles enter and exit the facility, the system continuously updates parking space availability, ensuring accurate information f
incoming users.

REQUIREMENT SPECIFICATIONS

The system analyst's primary goal is to comprehend the requirements of the new system that is being developed. The study of requirement specifications is critical for this. A preliminary survey of the existing system will be conducted in preparation for the development of the new system. It is investigated whether converting the system into an application program could solve the problems and eliminate the inefficiency of the existing system.

System requirement: -

Raspberry Pi

IR Sensor

Potentiometer

LCD Display

Software requirement: -

Proteus 8

MQTT (Media Query Telemetry Transport)

SYSTEM DESIGN

The basic goal of system design is to plan a solution for the problem. This phase is composed of several systems. This phase focuses on the detailed implementation of the feasible system. It emphasizes translating design specifications to performance specifications. System design has two phases of development logical and physical design.

During the logical design phase, the analyst describes inputs (sources), outputs (destinations), databases (data stores), and procedures (data flows) all in a format that meets the user requirements. The analyst also specifies the user needs at a level that virtually determines the information flow into and out of the system and the data resources. Here the logical design is done through data flow diagrams and database design.

The logical design is followed by physical design or coding. The physical design produces the working system by defining the design specifications, which tell the programmers exactly what the candidate system must do. The programmers write the necessary programs that accept input from the user, perform necessary processing on accepted data through call and produce the required report on a hard copy or display it on the screen.

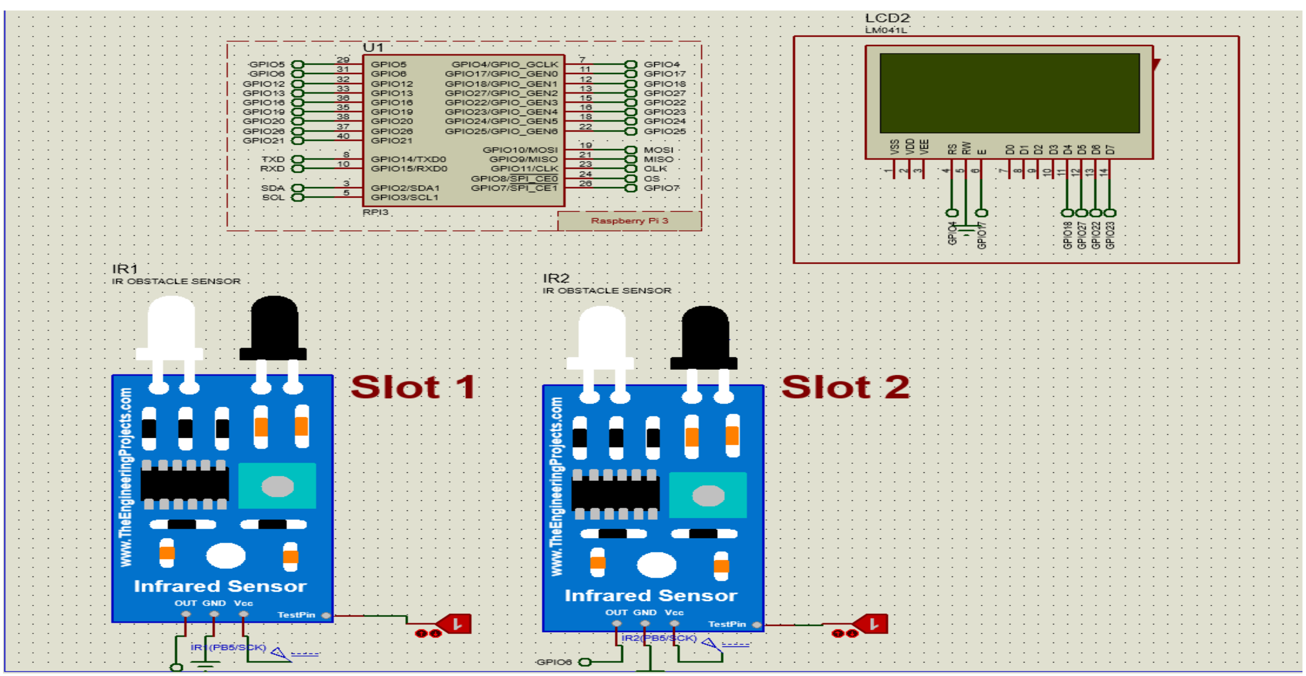

Fig 1.

PROJECT MODULES AND ITS DESCRIPTION

1) **Raspberry Pi :-** A powerful feature of the Raspberry Pi is the row of GPIO (general-purpose input/output) pins along the top edge of the board. A 40-pin GPIO header is found on all current Raspberry Pi boards, although it is unpopulated on Raspberry Pi Zero, Raspberry Pi Zero W, and Raspberry Pi Zero 2 W. The GPIO headers on all boards have a 0.1in (2.54mm) pin pitch. It is also known as Mini CPU.

2) **IR Sensor :-** An infrared sensor is a device that detects infrared radiation in its environment and outputs an electric signal. this sensor can detect movement as well as to measure the heat of an object. This sensor can detect infrared radiation, which is invisible to our eyes.

3) **LCD Display :-** LCD is utilized in a project to visualize the output of application. Liquid crystal displays (LCDs) have supplies which combine the properties of both liquids and crystals.

4) **Proteus 8 :-** Proteus 8 is a proprietary software tool suite used primarily for electronic design automation. It is developed by Lab Center Electronics Ltd and is available in multiple languages.

5) **MQTT :-** languages MQTT is a standard-based messaging protocol, or set of rules, used for machine-to-machine communication.

CODING

Here coding is given bellow: -

```
#!/usr/bin/python
import time
import RPi.GPIO as GPIO
import time
import os,sys
from urllib.parse import urlparse
import paho.mqtt.client as paho
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)

'''
define pin for lcd
'''
# Timing constants
E_PULSE = 0.0005
E_DELAY = 0.0005
delay = 1



# Define GPIO to LCD mapping
LCD_RS = 7
LCD_E  = 11
LCD_D4 = 12
LCD_D5 = 13
LCD_D6 = 15
LCD_D7 = 16
slot1_Sensor = 29
slot2_Sensor = 31
slot3_Sensor = 32
GPIO.setup(LCD_E, GPIO.OUT)  # E
GPIO.setup(LCD_RS, GPIO.OUT) # RS
GPIO.setup(LCD_D4, GPIO.OUT) # DB4
GPIO.setup(LCD_D5, GPIO.OUT) # DB5
GPIO.setup(LCD_D6, GPIO.OUT) # DB6
GPIO.setup(LCD_D7, GPIO.OUT) # DB7
GPIO.setup(slot1_Sensor, GPIO.IN)
GPIO.setup(slot2_Sensor, GPIO.IN)
GPIO.setup(slot3_Sensor, GPIO.IN)
# Define some device constants
```

```
LCD_WIDTH = 16    # Maximum characters per line
LCD_CHR = True
LCD_CMD = False
LCD_LINE_1 = 0x80 # LCD RAM address for the 1st line
LCD_LINE_2 = 0xC0 # LCD RAM address for the 2nd line
LCD_LINE_3 = 0x90# LCD RAM address for the 3nd line


def on_connect(self, mosq, obj, rc):
        self.subscribe("Fan", 0)

def on_publish(mosq, obj, mid):
    print("mid: " + str(mid))

mqttc = paho.Client()                              # object declaration
# Assign event callbacks
mqttc.on_connect = on_connect
mqttc.on_publish = on_publish

url_str = os.environ.get('CLOUDMQTT_URL', 'tcp://broker.emqx.io:1883')
url = urlparse(url_str)
mqttc.connect(url.hostname, url.port)

'''
Function Name :lcd_init()
Function Description : this function is used to initialized lcd by sending the
different commands
'''
def lcd_init():
  # Initialise display
  lcd_byte(0x33,LCD_CMD) # 110011 Initialise
  lcd_byte(0x32,LCD_CMD) # 110010 Initialise
  lcd_byte(0x06,LCD_CMD) # 000110 Cursor move direction
  lcd_byte(0x0C,LCD_CMD) # 001100 Display On,Cursor Off, Blink Off
  lcd_byte(0x28,LCD_CMD) # 101000 Data length, number of lines, font size
  lcd_byte(0x01,LCD_CMD) # 000001 Clear display
  time.sleep(E_DELAY)
'''
Function Name :lcd_byte(bits ,mode)
Fuction Name :the main purpose of this function to convert the byte data into bit
and send to lcd port
'''
def lcd_byte(bits, mode):
  # Send byte to data pins
  # bits = data
  # mode = True  for character
```

```
  #        False for command

  GPIO.output(LCD_RS, mode) # RS

  # High bits
  GPIO.output(LCD_D4, False)
  GPIO.output(LCD_D5, False)
  GPIO.output(LCD_D6, False)
  GPIO.output(LCD_D7, False)
  if bits&0x10==0x10:
    GPIO.output(LCD_D4, True)
  if bits&0x20==0x20:
    GPIO.output(LCD_D5, True)
  if bits&0x40==0x40:
    GPIO.output(LCD_D6, True)
  if bits&0x80==0x80:
    GPIO.output(LCD_D7, True)

  # Toggle 'Enable' pin
  lcd_toggle_enable()

  # Low bits
  GPIO.output(LCD_D4, False)
  GPIO.output(LCD_D5, False)
  GPIO.output(LCD_D6, False)
  GPIO.output(LCD_D7, False)
  if bits&0x01==0x01:
    GPIO.output(LCD_D4, True)
  if bits&0x02==0x02:
    GPIO.output(LCD_D5, True)
  if bits&0x04==0x04:
    GPIO.output(LCD_D6, True)
  if bits&0x08==0x08:
    GPIO.output(LCD_D7, True)

  # Toggle 'Enable' pin
  lcd_toggle_enable()
'''
Function Name : lcd_toggle_enable()
Function Description:basically this is used to toggle Enable pin
'''
def lcd_toggle_enable():
  # Toggle enable
  time.sleep(E_DELAY)
  GPIO.output(LCD_E, True)
```

```
  time.sleep(E_PULSE)
  GPIO.output(LCD_E, False)
  time.sleep(E_DELAY)
'''
Function Name :lcd_string(message,line)
Function  Description :print the data on lcd
'''
def lcd_string(message,line):
  # Send string to display

  message = message.ljust(LCD_WIDTH," ")

  lcd_byte(line, LCD_CMD)

  for i in range(LCD_WIDTH):
    lcd_byte(ord(message[i]),LCD_CHR)

lcd_init()
lcd_string("welcome ",LCD_LINE_1)
time.sleep(0.5)
lcd_string("Car Parking ",LCD_LINE_1)
lcd_string("System ",LCD_LINE_2)
time.sleep(0.5)
lcd_byte(0x01,LCD_CMD) # 000001 Clear display
# Define delay between readings
delay = 5

while 1:
  # Print out results
  rc = mqttc.loop()
  slot1_status = GPIO.input(slot1_Sensor)
  time.sleep(0.2)
  slot2_status = GPIO.input(slot2_Sensor)
  time.sleep(0.2)
  slot3_status = GPIO.input(slot3_Sensor)
  time.sleep(0.2)
  if (slot1_status == True):
   lcd_string("Slot1 Parked  ",LCD_LINE_1)
   mqttc.publish("slot1","1")
   time.sleep(0.2)
  else:
    lcd_string("Slot1 Free  ",LCD_LINE_1)
    mqttc.publish("slot1","0")
    time.sleep(0.2)
```

```
if (slot2_status == True):
 lcd_string("Slot2 Parked  ",LCD_LINE_2)
 mqttc.publish("slot2","1")
 time.sleep(0.2)
else:
  lcd_string("Slot2 Free  ",LCD_LINE_2)
  mqttc.publish("slot2","0")
  time.sleep(0.2)

if (slot3_status == True):
 lcd_string("Slot3 Parked  ",LCD_LINE_3)
 mqttc.publish("slot3","1")
 time.sleep(0.2)
else:
  lcd_string("Slot3 Free  ",LCD_LINE_3)
  mqttc.publish("slot3","0")
  time.sleep(0.2)
```

OUTPUT

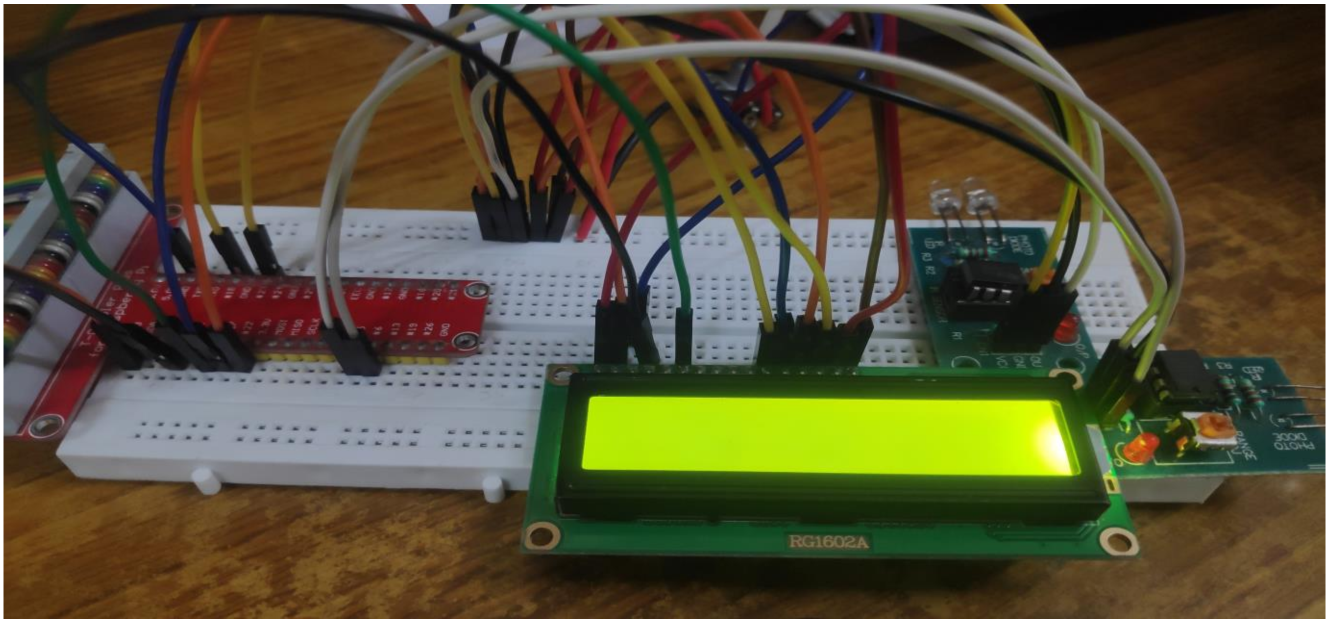

Fig 2.

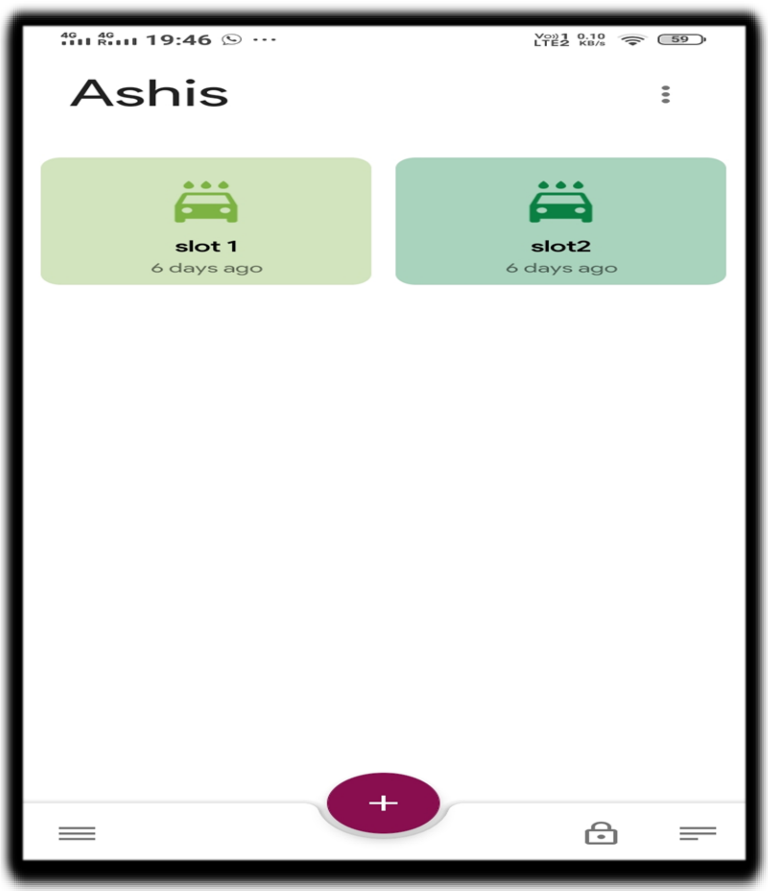

Fig 3.

FUTURE SCOPE

The future scope of car parking systems is promising, with potential advancements and expansions. As cities continue to embrace smart technologies, car parking systems can be integrated into broader smart city initiatives. This integration may involve connecting parking data with other urban infrastructure systems, such as traffic management, public transportation, and environmental monitoring, to create more efficient and sustainable urban environments. Future car parking systems may incorporate advanced features to further enhance the user experience. This could include personalized parking recommendations based on user preferences, seamless integration with navigation and ride-sharing apps, and the ability to reserve parking spaces in advance. By analyzing historical parking data and leveraging predictive analytics algorithms, future parking systems can anticipate demand patterns and optimize parking space allocation in real-time. This proactive approach can help mitigate congestion, reduce wait times, and improve overall efficiency. With the rise of alternative transportation modes such as electric scooters, bicycles, and shared mobility services, future car parking systems may evolve to accommodate these diverse transportation options. This could involve integrating parking facilities for multiple modes of transportation, as well as providing incentives for users to choose sustainable transportation methods. Advancements in autonomous vehicle technology may lead to the development of fully automated parking systems that enable vehicles to park themselves without human intervention. These systems could optimize parking space usage even further by reducing the need for designated driving lanes and maximizing parking density. Future car parking systems may prioritize environmental sustainability by incorporating features such as green parking infrastructure, electric vehicle charging stations, and incentives for carpooling and zero-emission vehicles. These initiatives can help reduce carbon emissions and promote eco-friendly transportation practices. With increasing concerns about vehicle theft and safety in parking facilities, future systems may integrate advanced security measures such as license plate recognition, video surveillance, and emergency alert systems to ensure the safety of both vehicles and users.

CONCLUSION

In conclusion, a meticulously designed and effectively implemented car parking system yields an array of advantages for both operators and users alike. Firstly, its meticulous design enables the optimization of space utilization through various sophisticated strategies. These strategies may encompass the utilization of compact car layouts, smart allocation algorithms, and the designation of specific parking zones tailored for different vehicle types. By strategically organizing parking spaces, the system ensures the efficient utilization of every square inch of available space, minimizing wastage and allowing for the accommodation of a greater number of vehicles within the same area. The system plays a crucial role in mitigating traffic congestion, a persistent challenge in urban environments. By furnishing real-time information regarding parking availability through digital displays, mobile applications, or similar mediums, it facilitates swift and efficient parking spot identification for drivers. This not only diminishes the frustration associated with hunting for parking but also curtails the number of vehicles circulating on the roads, thereby smoothing traffic flow and reducing overall congestion in densely populated areas. The system places a premium on enhancing the overall customer experience by integrating an array of user-friendly features. For instance, drivers benefit from instantaneous updates on parking availability, enabling them to plan their route and parking arrangements in advance. Online booking systems allow drivers to secure parking spaces ahead of time, eliminating the uncertainty of finding a spot upon arrival and ensuring peace of mind. Moreover, seamless payment options such as contactless payments or mobile wallet integration streamline the payment process, making it quick and hassle-free for users. The implementation of smart parking solutions is pivotal for addressing the evolving challenges of urban mobility and fostering sustainable urban development. As cities continue to expand and evolve, investing in innovative parking solutions becomes increasingly indispensable. By optimizing space utilization, reducing congestion, enhancing the customer experience, and contributing to a more seamless transportation ecosystem, these systems play a vital role in shaping the future of urban mobility and improving the quality of life for city dwellers.

REFERENCES

[1] Abhirup Khanna, Rishi Anand "IoT Based Smart Car Parking system "international conference on internet of things applications, IEEE conference publication, pp(266-270),Pune, India, 22Jan-24Jan,2016.

[2] Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP) (pp. 1-6). 2015, April.

[3] Rico, J., Sancho, J., Cendon, B., & Camus, M. "Parking easier by using context information of a smart city: Enabling fast search and management of parking resources". IEEE 27th

[4] Kafle, V. P., Fukushima, Y., &Harai, H. "ID-based communication for realizing IoT and M2M in future heterogeneous mobile networks".IEEE International conference on Recent

[5] Doukas, C., Capra, L., Antonelli, F., Jaupaj, E., Tamilin, A.,&Carreras,I. "Providing generic support for IoT and M2M for mobile devices. In Computing & Communication Technologies",. International Conference on Research, Innovation, and Vision for the Future (RIVF), (pp. 192-197),
2015 January.

[6] MuftahFraifer, Mikael fernstrom, "Smart Car Parking Prototype" IEEE third world forum on Internet of Things (WF-IoT) pp(649-654), 2016.