AUTOMATIC CAR PARKING WITH SMOKE DETECTION & BILL PAY SYSTEM

Submitted by:

Name: Md. Siful Islam	ID:EEE2103024126
Name: Md. Rahat Howlader	ID:EEE2103024124
Name: Md. Mahabub Rahman	ID:EEE2103024125
Name: Mohammad Naymul Haque	ID:EEE2002020004
Name:Md. Thachenul Haque	ID:EEE2103024066

Supervisor

Md. Ashikur Rahman

Lecturer

Department of EEE

Sonargaon University (SU)

Department of Electrical and Electronic Engineering

Sonargaon University (SU)

147/I, Panthapath,

Dhaka-1215, Bangladesh

Date of Submission: May, 2025

DECLARATION

We hereby, declare that the work presented in this project is the outcome of the investigation performed by us under the supervision of Md. Ashikur Rahman, Lecturer of the Electrical & Electronic Engineering Department, Sonargaon University. We also declare that no part of this project report and thereof has been or in being submitted elsewhere for the award of any degree.

We further undertake to identify the university against any loss or damage arising from breach of the forgoing obligation

Name ID	Name ID
Name ID	Name ID
Name ID	

CERTIFICATEION

This is to certify that the project report **"Automatic Car Parking with Smoke Detection & Bill Pay system"** is the confide record of the work done by-

For partial fulfillment of the requirement of their degree of B.Sc. in Electrical and Electronic Engineering at Sonargaon University.

The project report has been carried out under my guidance and is a confide record of the successful work. I wish them success in future.

Supervised By

Md. Ashikur Rahman
Lecturer
Department of EEE
Sonargaon University (SU)

ACKNOWLEDGEMENT

Firstly, we express our sincere thanks and gratitude to the almighty for his divine blessings which made it possible for us to complete this project successfully.

We are grateful and express our deepest gratitude to Md. Ashikur Rahman, Lecturer, Department of Electrical and Electronic Engineering, Sonargaon University. The deep knowledge and keen interest of our supervisor influenced me to carry out this project. His infinite patience, scholarly guidance, constant encouragement, constant energetic supervision, constructive criticism, valuable suggestions at all levels made this project possible.

The successful completion of this thesis would not have been possible without his constant inspiration and constant guidance. Hence we are extremely pleased to acknowledge our profound gratitude and respect to the maintainer.

We would like to express our gratitude to our Head of Department, as this project was completed under the Head of Department of Electrical and Electronics Engineering, Sonargaon University, Bangladesh.

We would like to thank our entire course mates from Sonargaon University, who participated in this discussion while completing the course work.

The author is grateful to all the respected faculty members of the Department of Electrical and Electronics Engineering for their cooperation and vital assistance in the successful completion of the thesis work.

ABSTRACT

The Automatic Car Parking System with Smoke Detection and Bill Payment is an innovative solution designed to enhance parking efficiency, safety, and convenience. This system integrates automated vehicle entry and exit, real-time smoke detection, and a digital payment mechanism to streamline parking operations. Utilizing sensors and IoT technology, vehicles are identified, and parking slots are allocated automatically. A smoke detection system continuously monitors the parking area for fire hazards, triggering alarms and notifications in case of emergencies. The integrated bill payment system allows seamless transactions via mobile apps or RFID-based cards, eliminating manual intervention. This smart parking system enhances safety, reduces congestion, and improves user experience by offering a hassle-free and secure parking environment. With its automated approach, the system minimizes human effort, optimizes space utilization, and ensures quick response to fire hazards, making it an essential advancement in modern urban infrastructure.

TABLE OF CONTENTS

Declaration ii
Certification iii
Acknowledgement iv
Abstract v
Table of Contant vi
List of Figures viii

CHAPTER 1 INTRODUCTION

1.1 Overview 1
1.2 Problem Statement 2
1.3 Objectives 2
1.4 Methodology 2
1.5 Work Flow Diagram 3
1.6 Research Outline 3

CHAPTER 2 LITERATURE REVIEW

2.1 Introduction 4
2.2 Literature Review 4
2.3 Summary 7

CHAPTER 3 SYSTEM ARCHITECTURE

3.1 Introduction 8
3.2 Block Diagram 8
3.3 Circuit Diagram 9
3.4 Working Principle 9
3.5 Cost Analysis 10

CHAPTER 4 HARDWARE & SOFTWARE ANALYSIS

4.1 Introduction 11
4.2 Hardware & Software 11
4.3 Arduino Nano 11

4.4 Node MCU 15
4.5 Switch Mode Power Supply (SMPS) 18
4.6 LCD Display 22
4.7 IR Sensor 24
4.8 Servo Motor 25
4.9 RFID Module 25
4.10 MQ2 Gas Sensor 27
4.11 Arduino IDE 29
4.12 Easy EDA 33

CHAPTER 5 RESULT ANALYSIS

5.1 Result Analysis 34
5.2 Complete Project Prtotype 35
5.3 Advantage 35
5.4 Application 35
5.5 Limitation 36
5.6 Discussion 36

CHAPTER 6 CONCLUSION

6.1 Conclusion 37
6.2 Future Scope 37

References 38
Appendix 40

LIST OF FIGURE

Figure No. **Page No**

1.1 Work Flow Diagram 3
3.1 Block Diagram of Our Project 8
3.2 Circuit Fiagram of Our Project 9
4.1 Arduino Nano 12
4.2 Arduino Nano Schematic Diagram 12
4.3 HowArduino Nano Looks Like 13
4.4 Microcontroller IC AT mega 328p 14
4.5 Node MCU 15
4.6 Node MCU Schematic Diagram 16
4.7 Node MCU pin Out 17
4.8 SMPS 18
4.9 SMPS Circuit 19
4.10 SMPS Circuit Connection 21
4.11 DC power Supply Way 22
4.12 16*2 LCD display 22
4.13 Pin Out of 16*2 LCD display 23
4.14 16*2 LCD display Interfacing with Arduino 23
4.15 IR Sensor 24
4.16 IR Sensor Reflection System 24
4.17 IR Sensor Detection System 24
4.18 Servo Motor 25
4.19 RFID Module 26
4.20 Schematic Diagram of RFID 26
4.21 MQ2 Gas Sensor 28
4.22 Arduino Software Interface IDE 30
4.23 Easy EDA Softwae Interface 33
5.1 Our Final Project 35

CHAPTER 1
INTRODUCTION

1.1 Overview

The Automatic Car Parking System with Smoke Detection and Bill Payment is a smart solution aimed at improving urban parking management. It automates vehicle entry and exit, detects fire hazards, and facilitates seamless digital payments. Using ultrasonic sensors, the system efficiently allocates parking spaces, reducing congestion and optimizing space utilization. A smoke detection module ensures safety by identifying fire risks and triggering alerts. The automated bill payment system, integrated with RFID or mobile payment, enhances convenience by eliminating manual transactions. This intelligent system enhances security, reduces human effort, and provides a hassle-free parking experience, making it ideal for modern smart cities.

These days, parking is a major issue in many public venues, including malls, multiplex systems, hospitals,workplaces, and market areas. The vehicle parking area has multiple lanes/slots for car parking. As a way to park an automobile one has to search for all of the lanes. Furthermore, a lot of investment and manual labour are required for this. Therefore, it is necessary to create an automated parking system that, at the entrance, clearly displays the availability of open parking spaces in any lane. It involves a system with a display outside the vehicle parking gate and an infrared transmitter-receiver pair in each lane. so that the person who wants to park their caris aware of the current state of parking space availability. The system uses an RFID-based bill payment for seamless transactions, while a display unit visually guides users to available parking slots. It automates check-in/check-out, saving time and optimizing parking efficiency.Traditional parking systems rely on security guards to keep an eye on the parking lots; they lack an intelligent monitoring system. Finding a spot to park takes up a lot of time and frequently causes traffic bottlenecks. Multiple parking lanes with multiple parking places in each lane exacerbate the situation. Using a parking management system would provide more comfort while reducing labor- intensive tasks and time. The display unit in the suggested system helps the user choose where to park their automobile by providing a visual depiction of the parking lot that includes both occupied and empty slots. The technology and software in the system would handle the cars' check-in and check-out procedures in addition to saving time.

1.2 Problem Statement

Traditional parking systems face challenges such as manual ticketing, payment delays, inefficient space utilization, and safety risks due to fire hazards. Manual parking management leads to congestion, increased waiting times, and security concerns. Additionally, the lack of a real-time monitoring system makes it difficult to detect fire or smoke incidents, posing risks to vehicles and occupants. Existing billing methods are often slow and inconvenient. To address these issues, an automated parking system with RFID-based bill payment and smoke detection is proposed. This system optimizes parking space usage, enhances safety with real-time alerts, and streamlines digital payments, ensuring a smooth and secure parking experience.

1.3 Objectives

The objectives of this project are:

- To study about Automatic Car Parking with Smoke Detection & Bill Pay system.
- To design and construct a Automatic Car Parking with Smoke Detection & Bill Pay system.
- To investigate maintain automatic the car slot.
- To develop an automated car parking system that efficiently manages vehicle entry, exit, and space allocation using sensors, reducing congestion and saving time.
- To integrate RFID-based bill payment and smoke detection for seamless transactions and enhanced safety, ensuring quick fire hazard detection and real-time alerts to prevent accidents in parking areas.
- To put the whole system into action so that we can observe its practical effects and verify our work.

1.4 Methodology

Our methodologies for the project:

- Creating an idea for Automatic Car Parking with Smoke Detection & Bill Pay system. And designing a block diagram & circuit diagram to know which components we need to construct it.
- Collecting all the components and programming the micro controller to control the whole system.
- Setting up all the components in a PCB board & then soldering. Lastly, assembling all the blocks in a board and to run the system & for checking purposes.

1.5 Work Flow Diagram

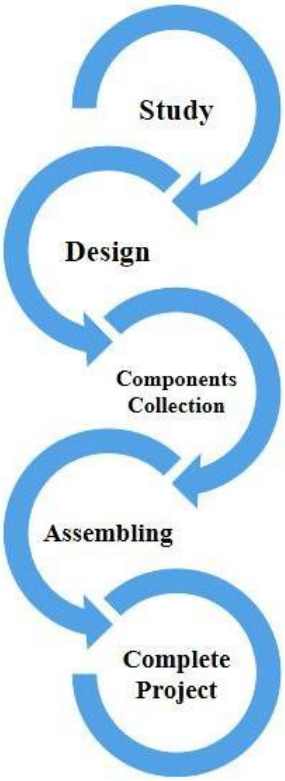


Figure 1.1: Working Flow Diagram

1.6 Research Outline

This project book consists of Six chapter. The first chapter contains the statement of the Introduction,Overview, Problem Statement, Objectives of the study ,Methodology and Work Flow Diagram,. Chapter two contains system Literatur e Review details. Chapter three Discussed in Block Diagram,Circuit Diagram,Working principle and Cost analysis.Chapter Four describes the hardware implementation with component details and the software which we have used for our work. Chapter Five deals with the result Analysis,shows the complete prototype,Advantage, Application and Limitation of the project that we have built. In the final chapter, we have discussed the , Future Scope, conclusion of the project.

CHAPTER 2
LITERATURE REVIEW

2.1 Introduction

In this chapter we discuss our system overview. In this paper they also work about **Automatic Car Parking with Smoke Detection & Bill Pay system**. In below section we will describe some previous literature review. From previous literature we gather more knowledge to build this project and we successfully made it .

2.1 Literature Review

S.Sunmathi [1] uses Infrared sensors to locate a car in parking spaces. The red LED colour glows when the slots are filled by a car. When there is no car filling the slots, a green LED light shine. The information of parking space is posted automatically on a website with the aid of the Arduino and WiFi module. When the user reaches the target at the car door the image is captured by the camera and the license plate number is taken with the help of the image processing and checked in a database with the recorded number plate. After testing, the parking space will be reserved, and the vehicle identifier will be displayed in the LCD and details will be posted on a website. When the vehicle leaves the parking fee for the gate is calculated by in-time and out-time calculations and the sum automatically charges from the account. It ensures that, before accessing the car parking system, the parking vacancies are identified on the Website that is accessible through the smartphone, tablet and laptop. A green LED for simple user identification indicates the empty slot and at the time of the departure, the user 's money is deducted automatically by measuring the time and length of the vehicle.

Zakria Qadir [2] uses Zig Bee technology which is much more efficient than Bluetooth and Wi-Fi. There are two main components of the wireless smart parking model; the first one is Arduino, whereas the second is a digital module. Kit interfaces Arduino to perform any task like other microsystems with multiple electrical components. It's relatively much simpler to program an Arduino board than the previous microcontroller because it receives and sends digitally data. It binds ZigBee, servo motor, IR sensors, LEDs and LCDs. GSM is incorporated into GPRS module because it can use high data rates of 172 Kbps and GSM is voice-centric so low data. GSM is also incorporated in GPRS mode.

Sanam Kazi [3] uses a Smart Parking App where user should install the mobile app on their mobile devices first. Constant Internet connection is necessary to access this system. When the application has been enabled, the user must open the smart parking app and will enter basic information, such as username, email address , mobile number, payment methods and vehicle details such as 2 wheelers and 4 Wheeler and Vehicle number plate data, when the user first visits the application. Such information will then be passed to the server by device, which will store and validate user data. Using GPS technology to map user position. The nearest parking areas are displayed to the user based on the user 's location. The parking area is selected, and the system displays the parking slots available and reserved, using a white and grey paint. To book this slot the user must click on the correct parking spot. The time during which the vehicle is parked is demanded by the user; the amount is paid according to the specified date. The intelligent parking system to minimize congestion of traffic is focused on the allocation of resources and the reservation of parking spaces for a fixed period. Upon successful completion of the booking timer, it is checked if the user exceeds the predefined time. If timer is longer than the predefined time, the user and the manager assigned for that area will be informed and charged the user for the exceeded time. If the user removes the slot before deadline the user is reimbursed for the remainder of the time. This enables the parking system to book the parking

Nazia Bibi [4] uses image processing where videos from 10 feet heighted camera were obtained from the top view of the car park. The system video data has been captured under various conditions and temporal shifts to increase the recognition capacity. Video is separated into frames. Then a key frame is extracted from each segment and further processing is applied to this key frame, in order to reduce computer complexity. When you enter the parking lot or exit the parking area with a radio operated toy car, motion of the vehicle is determined by the main frame subtraction. At first there were no parking lines in the parking arena. The parking area and vehicles intended for parking are entered manually by the driver. The device creates virtual parking lines automatically that take into account the vehicle size. In this training model, the maximum parking space capacity is 14. Each parking lot is issued with a specific numerical mark. The device will test the presence of the car in any block after the parking arena is divided into virtual blocks. On the image is applied a binary filter and a ROI region is used to remove car. Calculates the interest in ROI of the connected area and sets the threshold as reserved parking space to more than eighty. The number of blocks free of charge is shown in green for the divers and the blocks reserved in red.

Md Omar Hasan [5] uses ultrasonic sensors. When the parking space is empty, ultrasonic sensors change its value. Raspberry Pi 3 is linked to the sensors to get the data from a microcontroller computer. To collect data from the Raspberry Pi 3 using the CoAP protocol, Google Cloud IoT Core has been set up. Google Cloud stores and manages the value of empty parking spaces. You will see empty Cloud spaces by the user of a smartphone application. The user can book this place through the user interface. The Cloud parking space will be allocated to the user if the user books and the free parking space status will be changed. The smartphone application keeps the data on each user's license plate information. Deep learning framework for Google, TensorFlow is used for user authentication. The picture of the car license plate is captured in a real time scenario prior to entering the parking area by the Raspberry Pi 3 camera. The numbers of the license plate are then detected using the Convolutional Neural Networks (CNN). The user verification is validated and can reach the parking area if the data matches the custom dataset.

Development of the parking systems implemented in Europe, the United States and Japan is done with the incorporation of advanced technologies. Now-a- days, due to the rapid growth of the parking system, huge manpower is needed to locate the empty spaces in parking lots. Therefore, there is a need for an automatic or systematic parking system (Idris *et-al.*, 2009) [6]. Since, parking systems experience various challenges on a regular basis, therefore, various technologies have been used by the researchers to propose new systems. Various types of vehicle systems are implemented in the world like multi level parking system, volkswagen parking system, automated car parking system, Rotary car parking system etc. (Pashte*et-al.*, 2016) [7].

Recently, investigators use RFID based parking systems along with Infrared Sensors (IR). This system will facilitate faster user authentication and hence reduce waiting time and increase the efficiency of the parking space. This also discusses the Arduino Board (Sabnam*et-al.*, 2016) [8].

Similar study was done where discussion of the mechanical model and the software system related to the smart parking system has taken place. They also shared the hardware description about the microcontroller, relay and RFID tag reader. By using Proteus 8 software, practical development of software had been performed (Nimble *etal.*, 2016) [9].

Other than this, IoT technologies have been used to generate the smart parking system such that the driver can book a parking space from anywhere. This technology helps objects to be sensed and controlled remotely over existing networks. It even creates an opportunity for direct integration of the world into computer regions which improves efficiency (Karamchandani and choudhary. 2016) [10]. Their system consists of an IR sensor, RFID tag and E valet server. An android application is provided to drivers which provide the graphical view of parking spots (Osmani *et-al.,* 2016) [11]. Similar system is proposed in which users from rural areas can book a parking space by using a mobile app. They used sensors for detecting the vacant parking lots. They aim to lift the parking management from a purely physical system to computational service using IoT and Wi-Fi based technologies (Bachhav and Mechkul. 2017) [12]. Even using a cloud computing model, a low cost IoT based vehicle system is proposed that consists of HCSR-04 based ultrasonic sensors to detect the proximity of the car and status of occupancy of slots. Also, an IR sensor is deployed at the gates to sense the car number. A Blynk android app is also used to give notifications to different users about availability of slots (Mishra *et-al.,* 2019) [13].

Various researchers elaborate the research method of an automatic parking control system and designs a parking controller. They research the parking space scene recognition algorithm, and the results are experimentally verified. Path planning and a trajectory tracking simulation are performed according to the actual parking scenario. (Ma *et-al.,* 2017) [14]. Similarly, development of the parking system along with licence plate recognition, parking lot status and parking guidance system has taken place (Rashid *et-al.,* 2012) [15].

2.3 Summary

From the literature discussed above, we gained a lot of knowledge and we inspired to do this project. We were able to do it with everyone tireless work.

CHAPTER 3
SYSTEM ARCHITECTURE

3.1 Introduction

The project objectives, methods, literature evaluation, and other details were all clarified in the preceding chapter. The block diagram, circuit diagram, Working principle, and final project instrument cost analysis will all be covered in this chapter.

3.2 Block Diagram

Here is the block diagram of the **Automatic Car Parking with Smoke Detection & Bill Pay system** with all the essential components. All of the components are shown in below as a block in this diagram.Here we use Arduino Nano and Node MCU Micro-controller of our project. Other equipment's are attached with this micro-controller and work together for perform as our desire outcome.

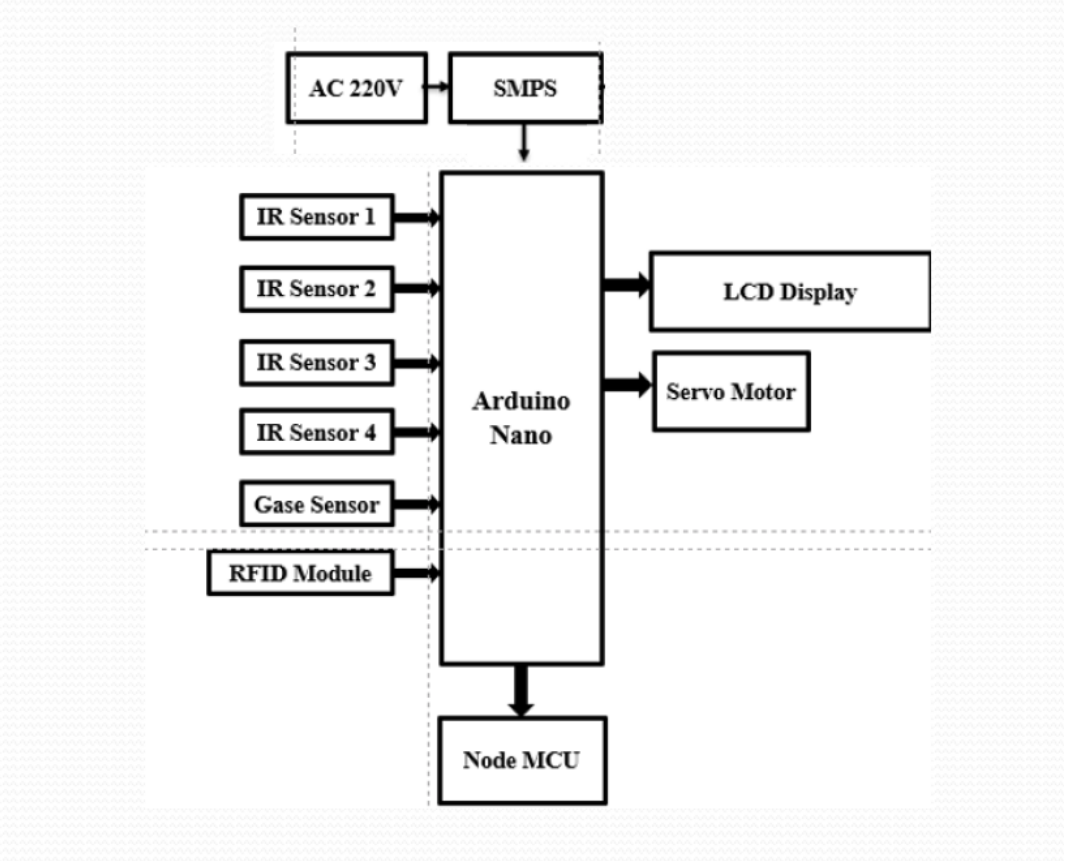


Figure 3.1: Block Diagram of Our Project

3.3 Circuit Diagram

In this part we show our project circuit design and connect out instrument through standard wire.

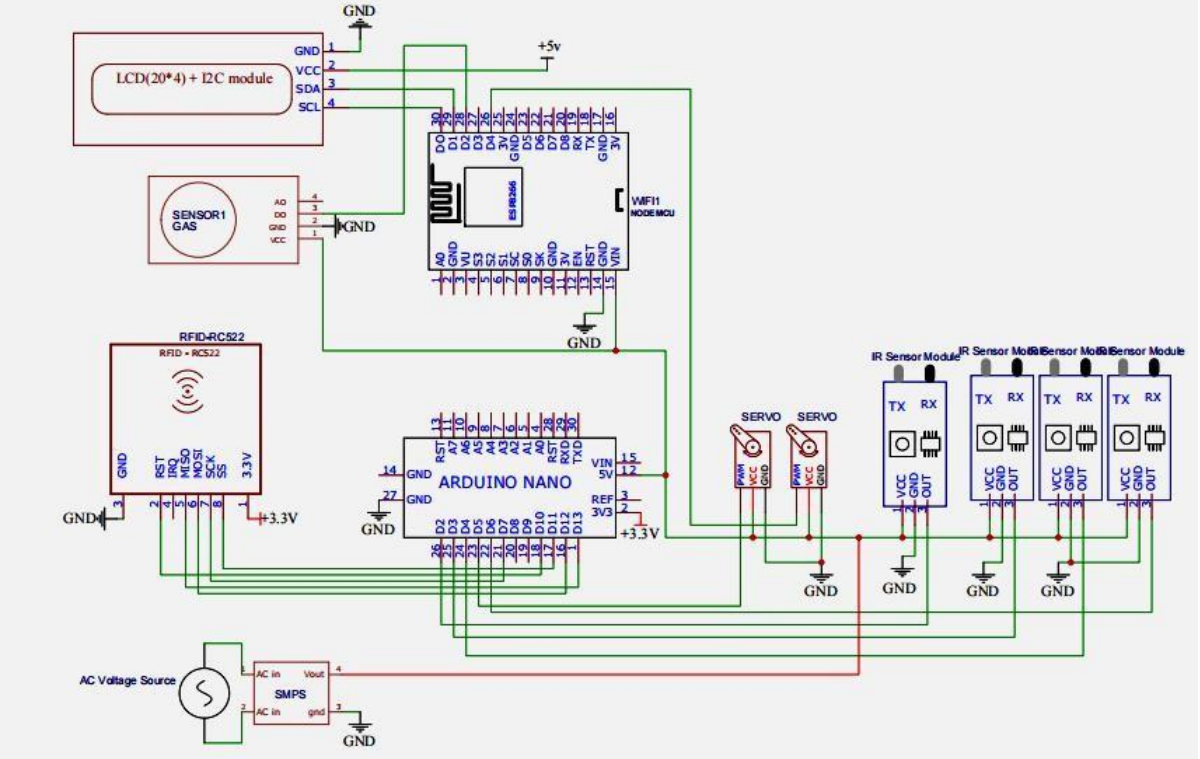


Figure 3.2: Circuit Diagram of Our Project

3.4 Working Principle

This IoT-based automatic car parking system efficiently manages vehicle entry, monitors parking slot availability, detects smoke, and facilitates bill payment.We used SMPS and rectifiers to power the entire project. At first SMPS convert AC to AC step down voltage & this voltage convert AC to DC via rectifier Circuit.

An IR sensor at the entrance detects an approaching car. If parking slots are available, the RFID module verifies the user's ID, and upon authentication, the servo motor opens the gate. Inside the parking lot, IR sensors track occupied and vacant slots, displaying the real-time status on an LCD display and updating the IoT system via NodeMCU.A gas sensor continuously monitors the environment for smoke or harmful gases. If smoke is detected, an alert is sent to the mobile app via NodeMCU, and safety measures can be activated.For billing, the RFID module logs entry and exit times, calculates the parking fee, and displays the amount on the LCD screen.

3.5 Cost Analysis

Table 1:List of Component with Price

Sl. no	Particulars	Specification	Qty.	Unit Price (Taka)	Total Price (Taka)
1	Arduino Nano	IC ATmega 328P	1	550	550
2	Node MCU	ESP32	1	600	600
3	SMPS	5V 7A	1	550	550
4	LCD Display	16*2	1	400	400
5	Gas Sensor	MQ2	1	300	300
6	RFID Module		1	350	350
7	IR Sensor		4	250	1000
8	Servo Motor	SG 90	2	250	500
9	Others				800
				Total	5150/=

CHAPTER 4

HARDWARE & SOFTWARE ANALYSIS

4.1 Introduction

In this section, we will discuss elaborately about "**Automatic Car Parking with Smoke Detection & Bill Pay system**" and the component description, features, working procedure of our all equipment.The system hardware fabricates composed of micro-controller unit, power unit, source unit, power store unite, sensor unit and many more related components.

4.2 Hardware & Software

Hardware

- Arduino Nano
- Node MCU
- SMPS
- LCD Display
- IR Sensor
- Servo Motor
- Gas Sensor
- RFID Module

Software

- Arduino Software
- Easy EDA

4.3 Arduino Nano

Arduino is open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments. Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling Lights, motors, and other actuators.

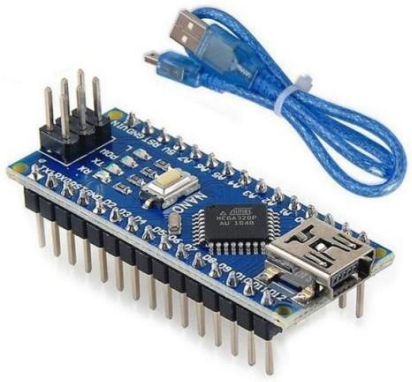


Figure 4.1: Arduino Nano

The micro-controller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software on running on a computer (e.g. Flash, Processing, Maxims'). Arduino Nano is a surface mount breadboard embedded version with integrated USB. It is a small, complete, and breadboard friendly component. It has everything that Decimal/ Duemilanove has (electrically) with more analog input pins and onboard +5V AREF jumper. Physically, it is missing power jack. The Nano can automatically sense and switch to the higher potential source of power.

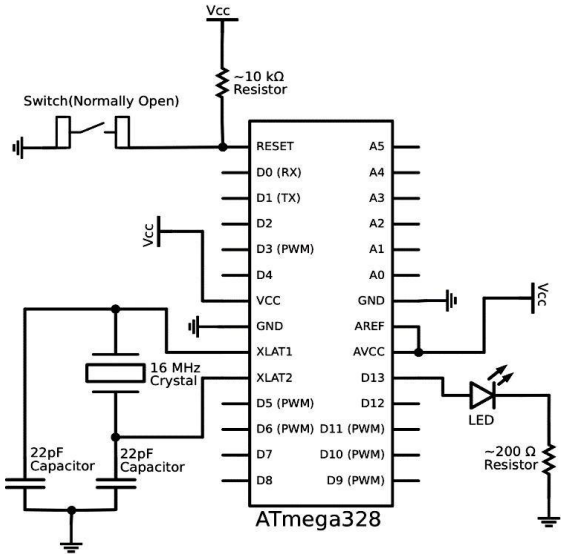


Figure 4.2: Arduino Nano Schematic Diagram

Nano’s got the breadboard-ability of the Boarding and the Minibus with smaller footprint than either, so users have more breadboard space. It’s got a pin layout that works well with the Mini or the Basic Stamp (TX, RX, ATN, and GND on one top, power and ground on the other). This new version 3.0 comes with ATMEGA328 which offer more programming and data memory space. It has two layers. That make it easier to hack and more affordable. One of the best features of Arduino Nano is, it’s easy to use, compact and also small.

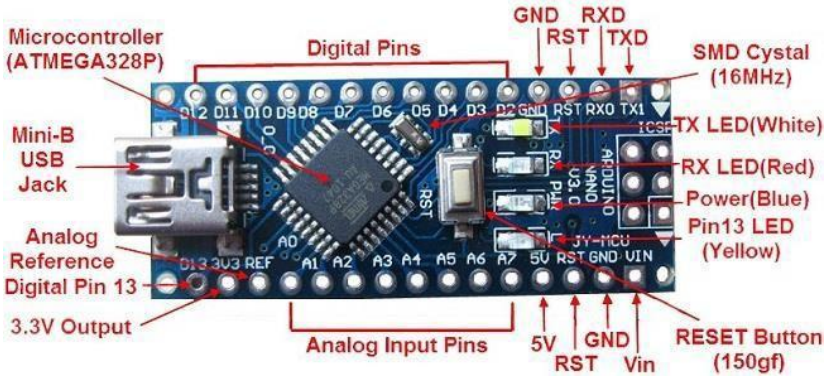


Figure 4.3: How Arduino Nano looks like

Specifications

- Micro-controller: Atmel ATmega328
- Operating Voltage (logic level): 5 V
- Input Voltage (recommended): 7-12 V
- Input Voltage (limits): 6-20 V
- Digital I/O Pins: 14 (of which 6 provide PWM output)
- Analog Input Pins: 8
- DC Current per I/O Pin: 40 mA
- Flash Memory: 32 KB (of which 2KB used by boot loader)
- SRAM : 2 KB
- EEPROM: 1 KB
- Clock Speed: 16 MHz
- Dimensions: 0.70” x 1.70”

Features

- Automatic reset during program download
- Power OK blue LED
- Green (TX), red (RX) and orange (L) LED
- Auto sensing/switching power input
- Small mini-B USB for programming and serial monitor
- ICSP header for direct program download
- Manual reset switch

Micro-controller IC ATmega328p



Figure 4.4: Micro-controller IC AT mega 328p

The high-performance Microchip Pico Power 8-bit AVR RISC-based micro-controller combines 32KB ISP flash memory with read-while-write capabilities, 1024B EEPROM, 2KB SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible timer/counters with compare modes, internal and external interrupts, serial programmable USART, a byte-oriented 2-wire serial interface, SPI serial port, a 6-channel 10-bit A/D converter (8-channels in TQFP and QFN/MLF packages), programmable watchdog timer with internal oscillator, and five software selectable power saving modes. The device operates between 1.8-5.5 volts. By executing powerful instructions in a single clock cycle, the device achieves throughput's approaching 1 MIPS per MHz, balancing power consumption and processing speed.

4.4 Node MCU

Node MCU is an open-source firmware for which open-source prototyping board designs are available. The name "Node MCU" combines "node" and "MCU" (micro-controller unit). The term "Node MCU" strictly speaking refers to the firmware rather than the associated development kits. Both the firmware and prototyping board designs are open source. The firmware uses the Lua scripting language. The firmware is based on the eLua project, and built on the Express if Non-OS SDK for ESP8266. It uses many open source projects, such as leucosin and SPIFFS. Due to resource constraints, users need to select the modules relevant for their project and build a firmware tailored to their needs. Support for the 32-bit ESP32 has also been implemented.

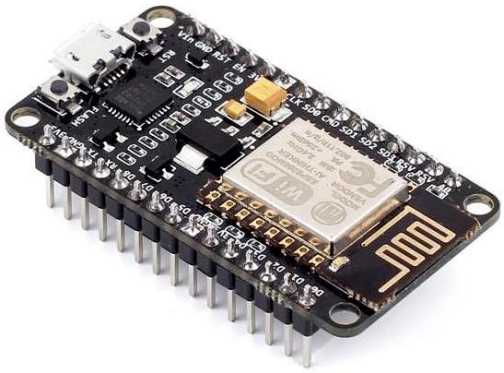


Figure 4.5: Node MCU

The prototyping hardware typically used is a circuit board functioning as a dual in-line package (DIP) which integrates a USB controller with a smaller surface-mounted board containing the MCU and antenna. The choice of the DIP format allows for easy prototyping on breadboards. The design was initially was based on the ESP-12 module of the ESP8266, which is a Wi-Fi SoC integrated with a TensilicaXtensa LX106 core, widely used in IoT applications.

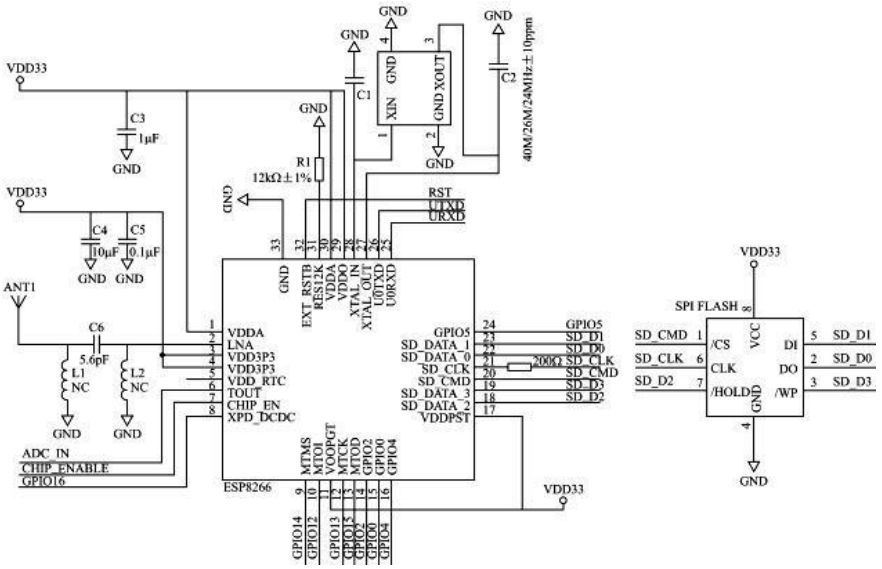


Figure 4.6: Node MCU Schematic Diagram

This an open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Express if Systems, and hardware which is based on the ESP-12 module. The term "Node MCU" by default refers to the firmware rather than the development kits. The firmware uses the Luascripting language. It is based on the eLua project, and built on the Espress if Non-OS SDK for ESP8266. Node MCU was created shortly after the ESP8266 came out. On December 30, 2013, Espressif Systems began production of the ESP8266. The ESP8266 is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core, widely used in IoT applications (see related projects). Node MCU started on 13 Oct 2014, when Hong committed the first file of nodemcu-firmware to GitHub. Two months later, the project expanded to include an open-hardware platform when developer Huang R committed the gerber file of an ESP8266 board, named devkit v0.9. Later that month, Tuan PM ported MQTT client library from Contiki to the ESP8266 SoC platform, and committed to Node MCU project, then Node MCU was able to support the MQTT IoT protocol, using Lua to access the MQTT broker. Another important update was made on 30 Jan 2015, when Devsaurus ported the u8glibto Node MCU project,[15] enabling Node MCU to easily drive LCD, Screen, OLED, even VGA displays. In summer 2015 the creators abandoned the firmware project and a group of independent contributors took over. By summer 2016 the Node MCU included more than 40 different modules. Due to resource constraints users need to select the modules relevant for their project and build a firmware tailored to their needs.

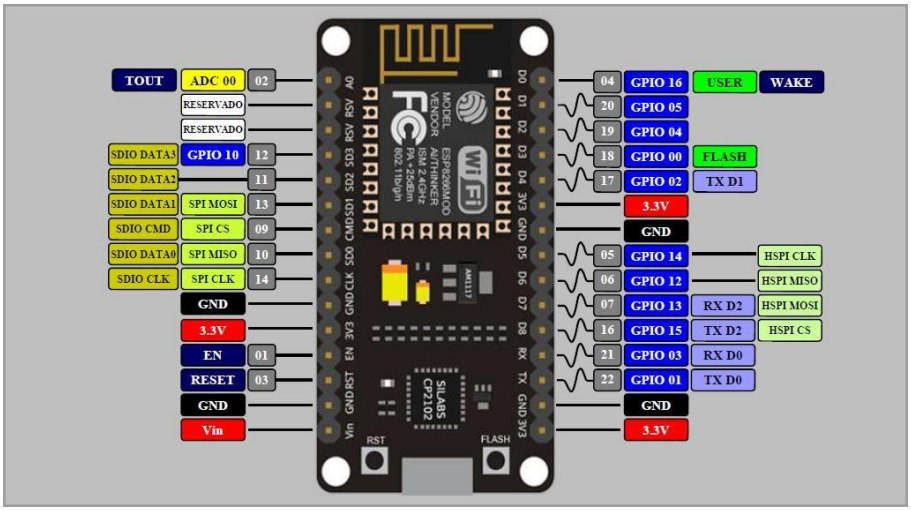


Figure 4.7: Node MCU Pin Out

Node MCU V3 ESP8266 ESP-12E is Wi-Fi development board that helps you to prototype your IoT product with few Lua script lines, or through Arduino IDE. The board is based on ESP8266 ESP-12E variant, unlike other ESP-12E, you won't need to buy a separate breakout board, USB to serial adapter, or even solder it to a PCB to get started, you will only need a USB cable (Micro USB).

Features:

- Communication interface voltage: 3.3V.
- Antenna type: Built-in PCB antenna is available.
- Wireless 802.11 b/g/n standard
- WiFi at 2.4GHz, support WPA / WPA2 security mode
- Support STA/AP/STA + AP three operating modes
- Built-in TCP/IP protocol stack to support multiple TCP Client connections (5 MAX)
- D0 ~ D8, SD1 ~ SD3: used as GPIO, PWM, IIC, etc., port driver capability 15mA
- AD0: 1 channel ADC
- Power input: 4.5V ~ 9V (10VMAX), USB-powered
- Current: continuous transmission: ≈70mA (200mA MAX), Standby: <200uA
- Transfer rate: 110-460800bps
- Support UART / GPIO data communication interface
- Remote firmware upgrade (OTA)
- Flash size: 4MByte.

4.5 Switch Mode Power Supply (SMPS)

A switched-mode power supply (switching-mode power supply, switch-mode power supply, switched power supply, SMPS, or switcher) is an electronic power supply that incorporates a switching regulator to convert electrical power efficiently. Like other power supplies, an SMPS transfers power from a DC or AC source (often mains power) to DC loads, such as a personal computer, while converting voltage and current characteristics. Unlike a linear power supply, the pass transistor of a switching-mode supply continually switches between low-dissipation, full-on and full-off states, and spends very little time in the high dissipation transitions, which minimizes wasted energy. A hypothetical ideal switched-mode power supply dissipates no power. Voltage regulation is achieved by varying the ratio of on-to-off time (also known as duty cycles). In contrast, a linear power supply regulates the output voltage by continually dissipating power in the pass transistor. This higher power conversion efficiency is an important advantage of a switched-mode power supply. Switched-mode power supplies may also be substantially smaller and lighter than a linear supply due to the smaller transformer size and weight.



Figure 4.8: SMPS

Switching regulators are used as replacements for linear regulators when higher efficiency, smaller size or lighter weight are required. They are, however, more complicated; their switching currents can cause electrical noise problems if not carefully suppressed, and simple designs may have a poor power factor. Switched-mode power supplies are classified according to the type of input and output voltages. The four major categories are:

- AC to DC
- DC to DC
- DC to AC
- AC to AC

A basic isolated AC to DC switched-mode power supply consists of:

- Input rectifier and filter
- Inverter consisting of switching devices such as MOSFETs
- Transformer
- Output rectifier and filter
- Feedback and control circuit

The input DC supply from a rectifier or battery is fed to the inverter where it is turned on and off at high frequencies of between 20 KHz and 200 KHz by the switching MOSFET or power transistors. The high-frequency voltage pulses from the inverter are fed to the transformer primary winding, and the secondary AC output is rectified and smoothed to produce the required DC voltages. A feedback circuit monitors the output voltage and instructs the control circuit to adjust the duty cycle to maintain the output at the desired level.

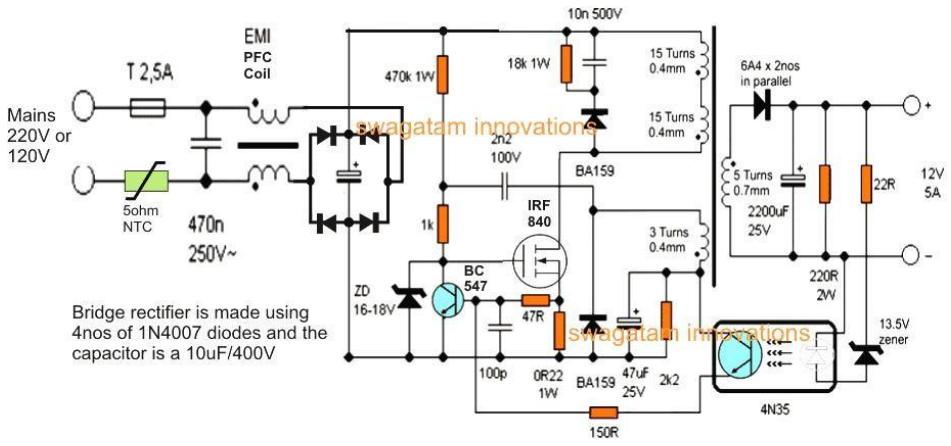


Figure 4.9: SMPS Circuit

Basic working concept of an SMPS

A switching regulator does the regulation in the SMPS. A series switching element turns the current supply to a smoothing capacitor on and off. The voltage on the capacitor controls the time the series element is turned. The continuous switching of the capacitor maintains the voltage at the required level.

Design basics

AC power first passes through fuses and a line filter. Then it is rectified by a full-wave bridge rectifier. The rectified voltage is next applied to the power factor correction (PFC) pre-regulator followed by the downstream DC-DC converter(s). Most computers and small appliances use the International Electrotechnical Commission (IEC) style input connector. As for output connectors and pin outs, except for some industries, such as PC and compact PCI, in general, they are not standardized and are left up to the manufacturer.

There are different circuit configurations known as topologies, each having unique characteristics, advantages and modes of operation, which determines how the input power is transferred to the output. Most of the commonly used topologies such as flyback, push-pull, half bridge and full bridge, consist of a transformer to provide isolation, voltage scaling, and multiple output voltages. The non-isolated configurations do not have a transformer and the power conversion is provided by the inductive energy transfer.

Advantages of switched-mode power supplies

- Higher efficiency of 68% to 90%
- Regulated and reliable outputs regardless of variations in input supply voltage
- Small size and lighter
- Flexible technology
- High power density
- Disadvantages:
- Generates electromagnetic interference
- Complex circuit design
- Expensive compared to linear supplies

Switched-mode power supplies are used to power a wide variety of equipment such as computers, sensitive electronics, battery-operated devices and other equipment requiring high

efficiency. Linear voltage IC regulators have been the basis of power supply designs for many years as they are very good at supplying a continuous fixed voltage output. Linear voltage regulators are generally much more efficient and easier to use than equivalent voltage regulator circuits made from discrete components such a zener diode and a resistor, or transistors and even op-amps.The most popular linear and fixed output voltage regulator types are by far the 78… positive output voltage series, and the 79… negative output voltage series.

These two types of complementary voltage regulators produce a precise and stable voltage output ranging from about 5 volts up to about 24 volts for use in many electronic circuits.There is a wide range of these three-terminal fixed voltage regulators available each with its own built-in voltage regulation and current limiting circuits. This allows us to create a whole host of different power supply rails and outputs, either single or dual supply, suitable for most electronic circuits and applications. There are even variable voltage linear regulators available as well providing an output voltage which is continually variable from just above zero to a few volts below its maximum voltage output.

Switch Mode Power Supply

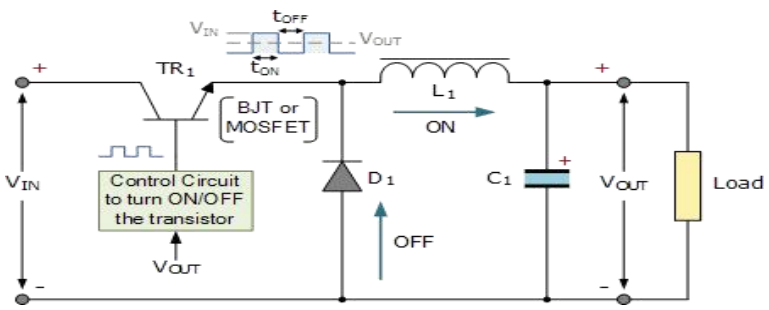


Figure 4.10: SMPS Circuit Connection

Most d.c. power supplies comprise of a large and heavy step-down mains transformer, diode rectification, either full-wave or half-wave, a filter circuit to remove any ripple content from the rectified d.c. producing a suitably smooth d.c. voltage, and some form of voltage regulator or stabiliser circuit, either linear or switching to ensure the correct regulation of the power supplies output voltage under varying load conditions. Then a typical d.c. power supply would look something like this:

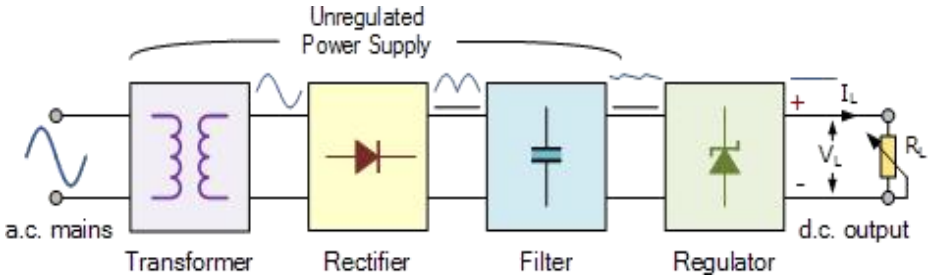


Figure 4.11: DC Power supply way

These typical power supply designs contain a large mains transformer (which also provides isolation between the input and output) and a dissipative series regulator circuit. The regulator circuit could consist of a single zener diode or a three-terminal linear series regulator to produce the required output voltage. The advantage of a linear regulator is that the power supply circuit only needs an input capacitor, output capacitor and some feedback resistors to set the output voltage.

4.6 LCD Display

The LCD (liquid crystal display) screen is an electronic display module and looks for various applications. A 16x2 LCD display is a very basic module and it is very commonly used in various devices and circuits. These modules are preferred over seven sections and many other segments.



Figure 4.12: 16*2 LCD Display

The reasons for having LCD are economic; Easily programmable, special and even custom characters (different in seven sections), there are no restrictions on displaying animations. A 16x2 LCD means it can display 16 characters per line and contains 2 lines. Each character on this LCD is displayed in a 5x7 pixel matrix.

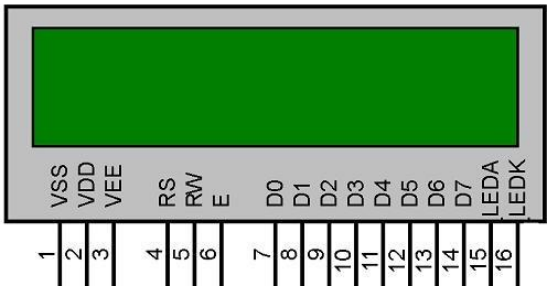


Figure 4.13: Pin out of 16*2 LCD Display

This LCD contains two articles called Command and Data. The command register stores the command instructions on the LCD. The command is a command given to the LCD to perform a predefined task such as starting it, clearing its screen, locating the cursor, controlling the display, etc.

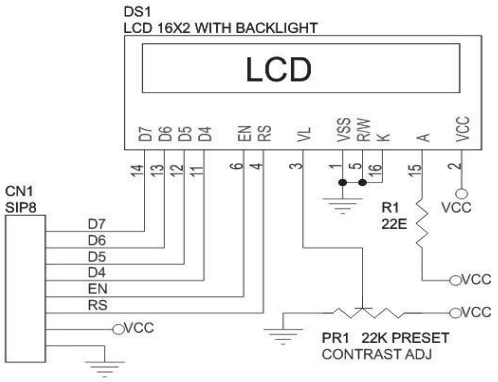


Figure 4.14: 16*2 LCD Display interfacing with Arduino

Feature

The features of this LCD mainly include the following.

- The operating voltage of this LCD is 4.7V-5.3V
- It includes two rows where each row can produce 16-characters.
- The utilization of current is 1mA with no backlight
- Every character can be built with a 5×8 pixel box
- The alphanumeric LCDs alphabets & numbers

4.7 IR Sensor

An infrared sensor is an electronics device that emits certain features around it and / or detects infrared radiation. Infrared sensors are capable of measuring the heat emitted by an object and detecting motion. In this project we will control the fan and light automatically through the closed loop system so the IR sensor for the system is definitely needed as a response. Here the fan will turn on and the light will turn on when the IR sensor detects an object and the fan and light will switch off automatically when an object leaves the house.



Figure 4.15: IR Sensor

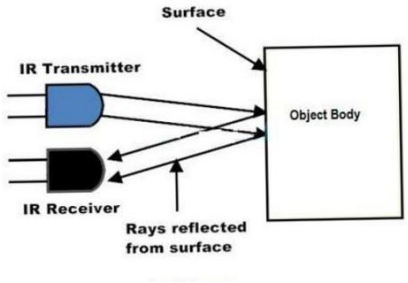


Figure 4.16: IR sensor reflection System.

Principles of operation

Sensor works with sense. When an object intersects the sensor light then the sensor detects something .Some infrared ray is deflected from the object and scene the length of this distance. In this sensor we can make a fixed length light and detect our exact object. If some object comes in front of this sensor then the sensor detects it and sends a signal in LED.

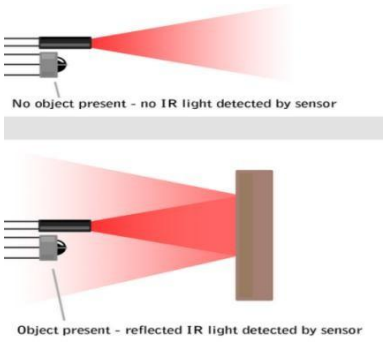


Figure 4.17: IR sensor Detection System

4.8 Servo Motor

A servo motor is a closed-loop servomechanism that uses position feedback to control its motion and final position. The input to its control is a signal (either analogue or digital) representing the position commanded for the output shaft. The motor is paired with some type of position encoder to provide position and speed feedback. In the simplest case, only the position is measured. The measured position of the output is compared to the command position, the external input to the controller. If the output position differs from that required, an error signal is generated which then causes the motor to rotate in either direction, as needed to bring the output shaft to the appropriate position. As the positions approach, the error signal reduces to zero and the motor stops.



Figure 4.18: Servo Motor

The very simplest servomotors use position-only sensing via a potentiometer and bang-bang control of their motor; the motor always rotates at full speed (or is stopped). This type of servomotor is not widely used in industrial motion control, but it forms the basis of the simple and cheap servos used for radio-controlled models. More sophisticated servomotors use optical rotary encoders to measure the speed of the output shaft and a variable-speed drive to control the motor speed. Both of these enhancements, usually in combination with a PID control algorithm, allow the servomotor to be brought to its commanded position more quickly and more precisely, with less overshooting.

4.9 RFID Module

RFID Module. RFID Reader Module, RC522 RFID RF IC card sensor module, RFID Module for Arduino, Barcode Reader, USB, USB Barcode Decoder. RFID stands for radio frequency identification which usually detects and locate particular object as commanded. perhaps without even being aware of it. At their simplest, RFID systems use tiny chips, called

"tags," to contain and transmit some piece of identifying information to an RFID reader, a device that in turn can interface with computers.



Figure 4.19: RFID Module

To begin understanding RFID, think of a conventional Point-of-Sale barcode reader scanning grocery barcodes. In its simplest form, an RFID system is much the same: it also can identify a package. However, unlike barcodes, RFID tags don't need a direct line of sight: within limits, we can now scan an unpacked skid of boxes. Next, think of RFID tags as mini databases, or as barcodes that can be written to, and that can accumulate information as they travel. At this point, RFID diverges qualitatively from bar coding, giving it great new potential. In an RFID system, RFID tags are "interrogated" by an RFID reader.

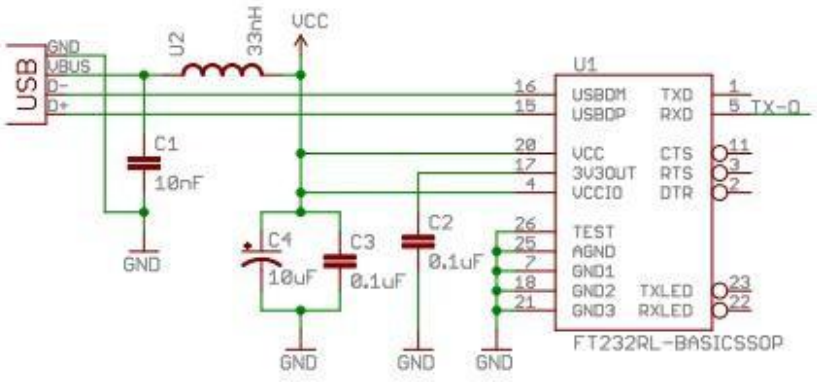


Figure 4.20: Schematic Diagram of RFID

The tag reader generates a radio frequency "interrogation" signal that communicates with the tags. The reader also has a receiver that captures a reply signal from the tags, and decodes that signal. The reply signal from the tags reflects, both literally and figuratively, the tag's data content. The reply signal is created as passive "backscatter" (to use the radio term) . *a.*

How RFID Works A Radio-Frequency Identification system has three parts: A scanning antenna A transceiver with a decoder to interpret the data A transponder - the RFID tag - that has been programmed with information. The scanning antenna puts out radio-frequency signals in a relatively short range.

The RF radiation does two things: It provides a means of communicating with the transponder (the RFID tag) AND It provides the RFID tag with the energy to communicate (in the case of passive RFID tags). This is an absolutely key part of the technology; RFID tags do not need to contain batteries, and can therefore remain usable for very long periods of time (maybe decades). The scanning antennas can be permanently affixed to a surface; handheld antennas are also available. They can take whatever shape you need; for example, you could build them into a door frame to accept data from persons or objects passing through. When an RFID tag passes through the field of the scanning antenna, it detects the activation signal from the antenna. That "wakes up" the RFID chip, and it transmits the information on its microchip to be picked up by the scanning antenna. In addition, the RFID tag may be of one of two types. Active RFID tags have their own power source; the advantage of these tags is that the reader can be much farther away and 119 still get the signal.

Even though some of these devices are built to have up to a 10-year life span, they have limited life spans. Passive RFID tags, however, do not require batteries, and can be much smaller and have a virtually unlimited life span. RFID tags can be read in a wide variety of circumstances, where barcodes or other optically read The read time is typically less than 100 milliseconds technologies are useless. The tag need not be on the surface of the object (and is therefore not subject to wear).

4.10 MQ2 Gas Sensor

The utility model can be used for gas leakage monitoring devices in families and factories, and is suitable for the detection of liquefied petroleum gas, butane, propane, methane, Hydrogen, smoke, etc. This is a very easy to use low cost semiconductor Gas sensor Module with analog and digital output.



Figure 4.21: MQ2 Gas Sensor

Features

- Adopt high quality double panel design, with power indication and TTL signal output indication.
- It has DO switch signal (TTL) output and AO analog signal output.
- TTL output valid signal is low level. When the output is low, the signal light is on, and the micro-controller or relay module can be directly connected.
- The analog output voltage increases with the concentration, the higher the voltage.
- It has better sensitivity to liquefied petroleum gas, natural gas, urban gas and smoke.
- MQ-2 MQ2 Smoke Gas LPG Butane Methane Sensor Detector Module
- With four screw holes, easy to locate.
- Product size: 32 (L), *20 (W), *22 (H)
- With long service life and reliable stability.
- Fast response recovery features

Specifications

- Input voltage: DC5V
- Power dissipation (current): 150mA
- DO output: TTL, numeric quantities 0 and 1 (0.1 and 5V)
- AO output: 0.1-0.3V (relatively pollution-free), the highest concentration of about 4V voltage
- Special reminder: after the sensor is energized, you need to preheat 20S or so, the data to be stable, sensor heating is a normal phenomenon, because the internal heating wire, if hot, it is not normal.

SOFTWARE

4.11 Arduino IDE

The digital microcontroller unit named as Arduino UNO can be programmed with the Arduino software IDE. There is no any requirement for installing other software rather than Arduino. Firstly, Select "Arduino UNO from the Tools, Board menu (according to the microcontroller on our board). The IC used named as ATmega328 on the Arduino UNO comes pre burned with a boot loader that allows us to upload new code to it without the use of an external hardware programmer.

Communication is using the original STK500 protocol (reference, C header files). We can also bypass the boot loader and programs the microcontroller through the ICSP (In Circuit Serial Programming) header. The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available. The ATmega16U2/8U2 is loaded with a DFU boot loader, which can be activated by:

On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2. On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.The Arduino UNO is one of the latest digital microcontroller units and has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL at (5V) with serial communication, which is available on digital pins 0 -(RX) for receive the data and pin no.1 (TX) for transmit the data. An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, an .in file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board.

The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial Communication on pins 0 and 1). A Software Serial library allows for serial communication on any of the Uno digital pins. The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus. Arduino programs are written

in C or C++ and the program code written for Arduino is called sketch. The Arduino IDE uses the GNU tool chain and AVR Lab to compile programs, and for uploading the programs it uses argued. As the Arduino platform uses Atmel microcontrollers, Atmel's development environment, AVR Studio or the newer Atmel Studio, may also be used to develop software for the Arduino.

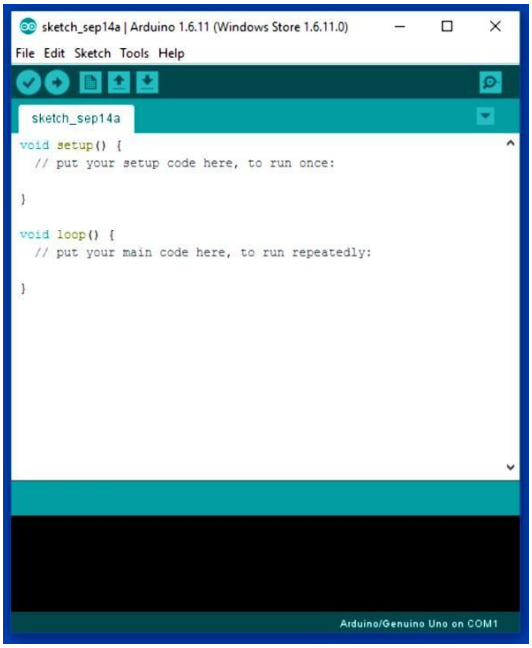


Figure 4.22: Arduino Software Interface IDE

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them.

Writing Sketches

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension .ino. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom righthand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

Sketchbook

The Arduino Software (IDE) uses the concept of a sketchbook: a standard place to store your programs (or sketches). The sketches in your sketchbook can be opened from the File > Sketchbook menu or from the Open button on the toolbar. The first time you run the Arduino software, it will automatically create a directory for your sketchbook. You can view or change the location of the sketchbook location from with the Preferences dialog. Beginning with version 1.0, files are saved with a .ino file extension. Previous versions use the .pde extension. You may still open .pde named files in version 1.0 and later, the software will automatically rename the extension to .ino.

Tabs, Multiple Files, and Compilation

Allows you to manage sketches with more than one file (each of which appears in its own tab). These can be normal Arduino code files (no visible extension), C files (.c extension), C++ files (.cpp), or header files (.h).

Uploading

Before uploading your sketch, you need to select the correct items from the Tools > Board and Tools > Port menus. The boards are described below. On the Mac, the serial port is probably something like /dev/tty.usbmodem241 (for an Uno or Mega2560 or Leonardo) or /dev/tty.usbserial-1B1 (for a Duemilanove or earlier USB board), or /dev/tty.USA19QW1b1P1.1 (for a serial board connected with a Keyspan USB-to-Serial adapter). On Windows, it's probably COM1 or COM2 (for a serial board) or COM4, COM5, COM7, or higher (for a USB board) - to find out, you look for USB serial device in the ports section of the Windows Device Manager. On Linux, it should be /dev/ttyACMx , /dev/ttyUSBx or similar. Once you've selected the correct serial port and board, press the upload button in the toolbar or select the Upload item from the Sketch menu. Current Arduino boards will reset automatically and begin the upload. With older boards (pre-Diecimila) that lack auto-reset, you'll need to press the reset button on the board just before starting the upload. On most boards, you'll see the RX and TX LEDs blink as the sketch is uploaded. The Arduino Software (IDE) will display a message when the upload is complete, or show an error.

When you upload a sketch, you're using the Arduino boot loader, a small program that has been loaded on to the microcontroller on your board. It allows you to upload code without using any additional hardware. The boot loader is active for a few seconds when the board resets; then it starts whichever sketch was most recently uploaded to the microcontroller. The bootloader will blink the on-board (pin 13) LED when it starts (i.e. when the board resets).

Libraries

Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data. To use a library in a sketch, select it from the Sketch > Import Library menu. This will insert one or more #include statements at the top of the sketch and compile the library with your sketch. Because libraries are uploaded to the board with your sketch, they increase the amount of space it takes up. If a sketch no longer needs a library, simply delete its #include statements from the top of your code.

There is a list of libraries in the reference. Some libraries are included with the Arduino software. Others can be downloaded from a variety of sources or through the Library Manager. Starting with version 1.0.5 of the IDE, you do can import a library from a zip file and use it in an open sketch. See these instructions for installing a third-party library.

Third-Party Hardware

Support for third-party hardware can be added to the hardware directory of your sketchbook directory. Platforms installed there may include board definitions (which appear in the board menu), core libraries, bootloaders, and programmer definitions. To install, create the hardware directory, then unzip the third-party platform into its own sub-directory. (Don't use "arduino" as the sub-directory name or you'll override the built-in Arduino platform.) To uninstall, simply delete its directory.

For details on creating packages for third-party hardware, see the Arduino IDE 1.5 3rd party Hardware specification.

Serial Monitor

This displays serial sent from the Arduino or Genuino board over USB or serial connector. To send data to the board, enter text and click on the "send" button or press enter. Choose the baud rate from the drop-down menu that matches the rate passed to Serial.begin in your sketch. Note that on Windows, Mac or Linux the board will reset (it will rerun your sketch)

when you connect with the serial monitor. Please note that the Serial Monitor does not process control characters; if your sketch needs a complete management of the serial communication with control characters, you can use an external terminal program and connect it to the COM port assigned to your Arduino board.

4.12 Easy EDA

Easy EDA is a web-based EDA tool suite that enables hardware engineers to design, simulate, share-publicly and privately-and discuss schematics, simulations and printed circuit boards. Other features include the creation of a bill of materials, Gerber files and pick and place files and documentary outputs in PDF, PNG and SVG formats. Easy EDA allows the creation and editing of schematic diagrams, SPICE simulation of mixed analogue and digital circuits and the creation and editing of printed circuit board layouts and, optionally, the manufacture of printed circuit boards.

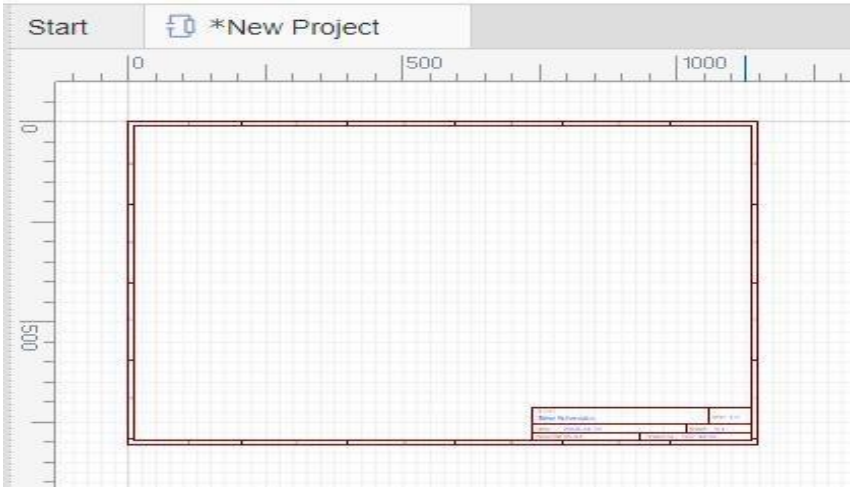


Figure 4.23: Easy EDA Software Interface

Subscription-free membership is offered for public plus a limited number of private projects. The number of private projects can be increased by contributing high quality public projects, schematic symbols, and PCB footprints and/or by paying a monthly subscription.Registered users can download Gerber files from the tool free of charge; but for a fee, Easy EDA offers a PCB fabrication service. This service is also able to accept Gerber file inputs from third party tools.

CHAPTER 5
RESULT ANALYSIS

5.1 Result Analysis

After finally completing this project, we ran it & we observed the output of this project. We can see that it is working well as expected. After making our project we observe it very careful. It works as we desire. Our project give output perfectly and all equipment are work perfectly. We check how much it works and we get perfect output from this project.

- Finally, we have completed our project successfully & check our project its run accurately according to our objective.
- At first, we start our system.
- Arduino Nano acts as the main controller for parking management, processing data from sensors and controlling motors.
- Node MCU Microcontroller Enables wireless connectivity for remote monitoring and bill payment features via Wi-Fi.
- IR sensors detect car presence; Servo motor opens/closes barriers based on availability.
- Gas sensor triggers an alert if smoke is detected, ensuring safety in case of a fire.
- Allows users to pay parking fees through the system, possibly integrated via NodeMCU and connected to a payment gateway.
- LCD displays parking status, available slots, fee calculation, and alerts (e.g., smoke or malfunction).

5.2 Complete Project Prototype

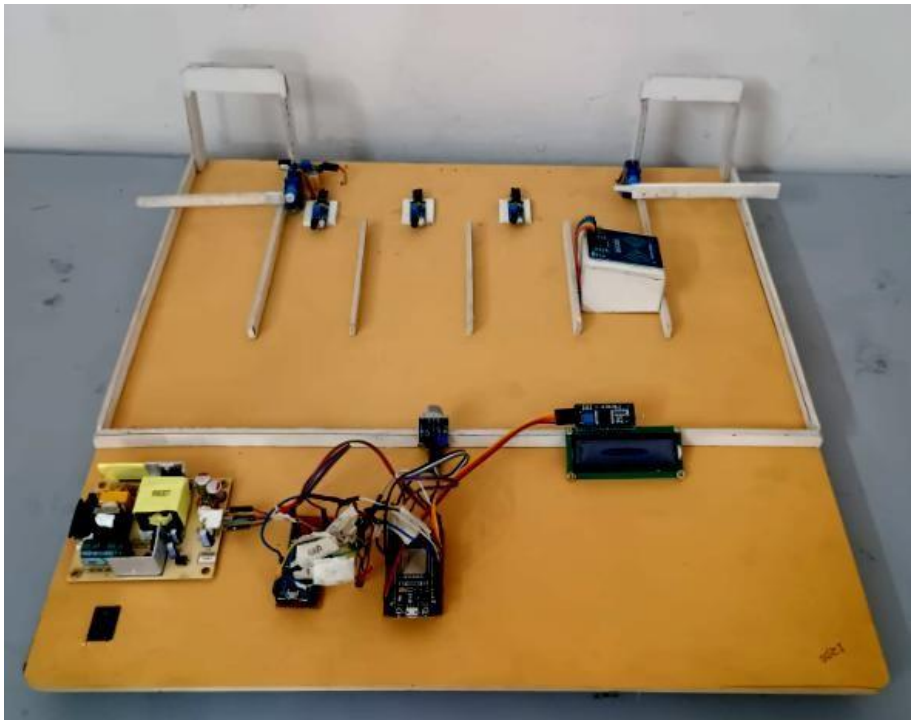


Figure 5.1:Our Final project

5.3 Advantage

There are many advantages of the project. Some of these are given below:

- Easy to use this system.
- User know the car limit's from outside of parking lot.
- Parking space can be used properly.
- Count Bill with the time.
- Smart Bill Pay System.
- Cost effective.

5.4 Application

This project has applications in many fields due its necessity. We have selected a few of them and they are given below:

- It can be used in any educational institutions
- It can be used in any industry or corporate offices
- It can be used in hospitals
- It can be used to identify parking lot space.
- It can be used in residential areas

5.5 Limitation

Through the project has many advantages it has also some limitation, they are given below:

- This project can now be only used for small scale purposes
- Used cheap Chinese products for the prototype so there's some processing delay present in the circuit.
- To get all notification mobile and project must be connected with the internet.
- No Energy Backup

5.6 Discussion

While working on our project, we did face some difficulties as it is a very complex system but the end results, we came up with were quite satisfactory. We have put the whole system through several tasks to validate our work and also have taken necessary notes for future improvements. Some future recommendations that we have involves improvement in system design and wiring, adding features for more efficient.

CHAPTER 6
CONCLUSION

6.1 Conclusion

The Automatic Car Parking with Smoke Detection & Bill Pay system provides an innovative and efficient solution for managing parking spaces while ensuring safety and convenience. By integrating key technologies such as the Arduino Nano and NodeMCU microcontrollers, the system automates vehicle entry and exit with the help of IR sensors and servo motors. The use of an LCD display offers real-time feedback on parking availability, status, and fees, enhancing user experience. The inclusion of a gas sensor for smoke detection adds an important layer of safety, alerting users and authorities in case of fire or gas leakage, thus ensuring the security of the parking facility. Additionally, the RFID module streamlines user authentication and parking fee payment, further simplifying the process. The system is powered by a reliable SMPS, ensuring stable operation of all components. Overall, this project combines automation, safety, and ease of use, making it a highly effective and smart solution for modern car parking management in both residential and commercial settings.

6.2 Future Scope

As we have already discussed about the limitations of our project so definitely there's room for improvement and thus, we have lots of future scope of work available to us for this project. Some of these are listed below:

- In future development this project can be develop by more sensor and alarm System
- In future, I will try it to apply in real field.
- In Future develop add extra power system.
- In the future, add a GSM module to send an SMS alert system.
- In future we will add camera monitoring system in Parking Slot.

REFERENCE

[1] S. Sunmathi, M. Sandhya, M. Sumitha and A. Kirthika, "Smart Car Parking using Image Processing", 2019 5th International Conference on Advanced Com- puting and Communication Systems.

[2] Zakria Qadir, Fadi Al-Turjman, Muazzam A. Khan and Tayfun Nesimoglu," ZIGBEE Based Time and Energy Efficient Smart Parking System Using IoT", 2018 18th Mediterranean Microwave Symposium.

[3] Sanam Kazi, Shirgaonkar Nuzat, Ansari Nashrah and Qureshi Rameeza, "Smart Parking System to Reduce Traffic Congestion", 2018 International Conference on Smart City and Emerging Technology.

[4] Nazia Bibi, Muhammad Nadeem Majid, Hassan Dawood and Ping Guo, "Auto- matic Parking Space Detection System", 2017 2nd International Conference on Multimedia and Image Processing.

[5] Md Omar Hasan, Md. Motaharul Islam and Yazed Alsaawy, "Smart Parking Model based on Internet of Things (IoT) and TensorFlow", 2019 7th International Conference on Smart Computing and Communications.

[6] Idris M.Y.I., Y.Y. Leng, E.M. Tamil, N.M. Noor *et-al.* (2009) 'Car Park System: A Review of Smart Parking System and its Technology' *Information Technology Journal,* Vol- 8, No.-2, pp- 101-113

[7] Pashte P., V. Narkhede,. S. Nade*et-al* (2016). 'Design and Analysis of Rotary Automated Car Parking System'. *International Journal for Scientific Research & Development*| Vol-4, No.4, pp- 28-31

[8] SabnamMasiha, M. Das, and P.A. Kashyap. (2016). 'Automatic Car Parking System'*ADBU-Journal of Engineering Technology, AJET,* Vol- 4, No.-1, pp-120-122

[9] Nimble J., P. Bhegade, S. Surve*et-al.*(2016) 'Automatic Smart Car Parking System'.*International Journal of Advances In Electronics And Computer Science*, Vol-3 No-3, pp- 49-51

[10] Karamchandani S., S. Choudhary. (2016). 'Research Paper on Multilevel Parking'. *International Journal of Engineering Sciences & Research Technology*, Vol-5, No.-12, pp-564-565

[11] Osmani A.H., A. Gawade, M. Nikam*et-al.* (2016). 'Research paper on Smart City Parking System'. *IJARIIE*, Vol-2, No.-3, pp-2998-3000

[12] Bachhav J.D, and Mechkul M. A. (2017).'Smart Car Parking System', *International Research Journal of Engineering and Technology, IRJET* , Vol- 4, No.-6, pp-3036-3038

[13] Mishra A. , A. Karmakar, A. Ghatak*et-al.* (2019). 'Low Cost Parking System For Smart Cities: A Vehicle Occupancy Sensing And Resource Optimization Technique Using IoT And Cloud PaaS'. *International Journal of Scientific & Technology Research.*, Vol-8, No.-9, pp-115-122

[14] Ma S., H. Jiang, M. Han, J. Xieet-al.(2017). 'Research on Automatic Parking Systems Based on Parking Scene Recognition' *IEEEAccess*, Vol-5,pp- 21901-21917

[15] Rashid M.M., A. Musa, M. Ataur Rahman, et-al.(2012). 'Automatic Parking Management System and Parking Fee Collection Based on Number Plate Recognition', *International Journal of Machine Learning and Computing*, Vol-2, No.-2, pp- 93-98

APPENDIX

Microcontroller Programme : Arduino

```
#include <SPI.h>
#include <MFRC522.h>
#include <Servo.h>

#define SS_PIN 10
#define RST_PIN 9
int card_pin = 6;
int ring_pin = 7;
int servo_pin = 4; // Change this to the pin connected to your servo
int servo_pin1 = 5; // Change this to the pin connected to your servo
int button=2;
int btn;
#define card "B3 61 DF 26"//A7 27 6C 1C
#define tag "E6 85 B2 F8"//F9 1D D0 CF


MFRC522 mfrc522(SS_PIN, RST_PIN); // Create MFRC522 instance.
Servo myServo;
Servo myServo1;

void setup()
{
  Serial.begin(9600); // Initiate a serial communication
  SPI.begin();        // Initiate SPI bus
  mfrc522.PCD_Init(); // Initiate MFRC522
  Serial.println("Approximate your card to the reader...");
  Serial.println();
  pinMode(card_pin, OUTPUT);
  pinMode(ring_pin,  OUTPUT);
  pinMode(button, INPUT);

  myServo.attach(servo_pin); // Attach the servo to the specified pin
  myServo1.attach(servo_pin1); // Attach the servo to the specified pin

   myServo.write(125);
     myServo1.write(125);

}

void loop()
{
//  btn=digitalRead(button);
// Serial.println(btn);
```

```
//if(btn==1){
//    digitalWrite(card_pin, HIGH);
//    myServo.write(70); // Rotate the servo to 90 degrees //decrease to rotate upwards more
//    myServo1.write(70); // Rotate the servo to 90 degrees //decrease to rotate upwards more
//    delay(3000);
//    myServo.write(125); // Return the servo to 0 degrees
//    myServo1.write(125); // Return the servo to 0 degrees
//    digitalWrite(card_pin, LOW);
//  }
//if(btn==0){
  automatic();
//  }
}

void automatic(){
  // Look for new cards
  if (!mfrc522.PICC_IsNewCardPresent())
  {
    return;
  }
  // Select one of the cards
  if (!mfrc522.PICC_ReadCardSerial())
  {
    return;
  }
  // Show UID on the serial monitor
  Serial.print("UID tag :");
  String content = "";
  byte letter;
  for (byte i = 0; i < mfrc522.uid.size; i++)
  {
    Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
    Serial.print(mfrc522.uid.uidByte[i], HEX);
    content.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " "));
    content.concat(String(mfrc522.uid.uidByte[i], HEX));
  }
  Serial.println();
  Serial.print("Message : ");
  content.toUpperCase();
  if (content.substring(1) == card) // Change here the UID of the card/cards that you want to
give access
  {
    Serial.println("UID: " + content);
    digitalWrite(card_pin, HIGH);
    myServo.write(100); // Rotate the servo to 90 degrees //decrease to rotate upwards more
    myServo1.write(70); // Rotate the servo to 90 degrees //decrease to rotate upwards more
    delay(3000);
    myServo.write(0); // Return the servo to 0 degrees
    myServo1.write(125); // Return the servo to 0 degrees
    digitalWrite(card_pin, LOW);
```

```
 }
 else if (content.substring(1) == tag)
 {
  Serial.println("UID: " + content);
  digitalWrite(ring_pin, HIGH);
  delay(3000);
  digitalWrite(ring_pin, LOW);
 }
 else
 {
  Serial.println(" Access denied");
  Serial.println("UID: " + content);
  digitalWrite(ring_pin, HIGH);
  delay(3000);
  digitalWrite(ring_pin, LOW);
 }
 }
```

Microcontroller Programme : Node MCU

```
#include <WiFi.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <ESP32Servo.h>  // Correct library for ESP32 Servo control

// IR sensor pins
#define slot1Pin 5
#define slot2Pin 18
#define slot3Pin 19
#define gateIRPin 34  // IR sensor for gate detection
#define servoPin 4  // Pin for controlling the servo
#define smokePin 35  // Smoke sensor analog pin (MQ-2 or similar)

// Wi-Fi credentials
const char *ssid = "ParkingSystem";
const char *password = "12345678";

// LCD: (Address 0x27 might be 0x3F depending on your module)
LiquidCrystal_I2C lcd(0x27, 16, 2);

WiFiServer server(80);

Servo gateServo;  // Servo object for controlling the gate

// Threshold for smoke detection
const int smokeThreshold = 400;  // Adjust this value based on your sensor's calibration

void setup() {
  Serial.begin(115200);
```

```
  pinMode(slot1Pin, INPUT);
  pinMode(slot2Pin, INPUT);
  pinMode(slot3Pin, INPUT);
  pinMode(gateIRPin, INPUT);

  lcd.init();
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("Parking System");
  lcd.setCursor(0, 1);
  lcd.print("Initializing...");
  delay(2000);

  WiFi.softAP(ssid, password);
  Serial.println("Access Point Started");
  Serial.print("IP Address: ");
  Serial.println(WiFi.softAPIP());

  gateServo.attach(servoPin, 500, 2400); // Servo initialization for ESP32
  gateServo.write(80); // Set the gate to the closed position initially

  server.begin();
}

void loop() {
  // Read slot status
  bool slot1 = digitalRead(slot1Pin) == LOW;
  bool slot2 = digitalRead(slot2Pin) == LOW;
  bool slot3 = digitalRead(slot3Pin) == LOW;
  bool gateIR = digitalRead(gateIRPin) == LOW; // Detect if the gate IR is triggered

  // Read the smoke sensor value
  int smokeValue = analogRead(smokePin);
  bool smokeDetected = smokeValue < smokeThreshold;

  // Update LCD
  lcd.setCursor(0, 0);
  lcd.print("S1:");
  lcd.print(slot1 ? "Busy " : "Free");
  lcd.print(" S2:");
  lcd.print(slot2 ? "Busy " : "Free");

  lcd.setCursor(0, 1);
  lcd.print("S3:");
  lcd.print(slot3 ? "Busy " : "Free ");
  lcd.print("       "); // Clear remaining chars

  // Display smoke detection warning
  if (smokeDetected) {
```

```
  lcd.setCursor(9, 1);
  lcd.print("Smoke !!!  ");
  // Add safety measures here if necessary, like triggering alarms
 }

// Check if all slots are full
if (slot1 && slot2 && slot3) {
  if (gateIR) {
    // Gate stays closed if slots are full
    gateServo.write(80);  // Close gate
  }
} else {
  if (gateIR) {
    gateServo.write(0);  // Open gate (80 degrees)
    delay(2500);
  } else {
    gateServo.write(80);   // Keep gate closed if no object detected
  }
}

// Handle client (web page)
WiFiClient client = server.available();
if (client) {
  Serial.println("Client Connected");
  String request = client.readStringUntil('\r');
  client.flush();

  if (request.indexOf("/status") != -1) {
    String json = "{";
    json += "\"slot1\":\"" + String(slot1 ? "Occupied" : "Free") + "\",";
    json += "\"slot2\":\"" + String(slot2 ? "Occupied" : "Free") + "\",";
    json += "\"slot3\":\"" + String(slot3 ? "Occupied" : "Free") + "\",";

    // Add smoke status to the JSON response
    json += "\"smoke\":\"" + String(smokeDetected ? "Detected" : "Clear") + "\"";
    json += "}";

    client.println("HTTP/1.1 200 OK");
    client.println("Content-Type: application/json");
    client.println();
    client.print(json);
    client.stop();
    Serial.println("Client Disconnected");
    return;
  }
  // Default HTML page response
  String html = "<!DOCTYPE html><html><head><title>Parking Slots</title>";
  html += "<style>";
  html += "body{font-family:sans-serif;text-align:center;background:#f5f5f5}";
```

```
  html += ".slot{display:inline-block;width:150px;height:150px;margin:20px;font-
size:20px;border-radius:10px;position:relative;transition:0.3s ease;}";
  html += ".free{background:#4CAF50;color:black;}";
  html += ".occupied{background:#FFFF00;color:black;}";
  html += ".car{position:absolute;top:40px;left:0;right:0;font-
size:50px;opacity:0;animation:fadeIn 0.8s forwards;}";
  html += "@keyframes fadeIn{from{opacity:0; transform:translateY(20px);}to{opacity:1;
transform:translateY(0);}}";
  html += ".notification{color:white;padding:10px;background-color:red;margin-
top:20px;border-radius:5px;display:none;}";
  html += "</style>";
  html += "<script>";
  html += "function updateStatus(){";
  html += "fetch('/status').then(r=>r.json()).then(data=>{";
  html += "updateSlot('slot1', data.slot1);";
  html += "updateSlot('slot2', data.slot2);";
  html += "updateSlot('slot3', data.slot3);";
  html += "document.getElementById('smokeStatus').innerText = 'Smoke: ' + data.smoke;";
  html += "if(data.smoke ===
'Detected'){document.getElementById('smokeNotification').style.display =
'block';}else{document.getElementById('smokeNotification').style.display = 'none';}";
  html += "});}";
  html += "function updateSlot(id, status){";
  html += "let slot = document.getElementById(id);";
  html += "slot.className = 'slot ' + (status === 'Occupied' ? 'occupied' : 'free');";
  html += "slot.innerHTML = id.replace('slot','Slot ') + '<br>' + status;";
  html += "if(status === 'Occupied'){slot.innerHTML += \"<div class='car'>  </div>\";}";
  html += "}";
  html += "setInterval(updateStatus, 1000);";
  html += "window.onload = updateStatus;";
  html += "</script></head><body>";
  html += "<h2>    Smart Parking Monitor</h2>";
  html += "<div id='slot1' class='slot'>Slot 1<br>Loading...</div>";
  html += "<div id='slot2' class='slot'>Slot 2<br>Loading...</div>";
  html += "<div id='slot3' class='slot'>Slot 3<br>Loading...</div>";
  html += "<div id='smokeStatus'>Smoke: Clear</div>";  // Display smoke status on the
webpage
  html += "<div id='smokeNotification' class='notification'>  Smoke Detected! Please take
action immediately.</div>";
  html += "</body></html>";

  client.println("HTTP/1.1 200 OK");
  client.println("Content-type:text/html");
  client.println();
  client.println(html);
  client.stop();
  Serial.println("Client Disconnected");
 }

 delay(500); // slight delay to avoid LCD flickering}
```