



# **SMART PARKING SYSTEM USING AURDINO WITH NODEMCU**

**1Y PRANAV SAI KUMAR REDDY, 2CH BHARGAV, 3G YESHWANTH KUMAR, 4Mr.  
Rajkumar D Bhure**

**1STUDENT, 2STUDENT , 3STUDENT, 4ASSOCIATE PROFESSOR**

**1J B INSTITUTE OF ENGINEERING AND TECHNOLOGY,**

**2J B INSTITUTE OF ENGINEERING AND TECHNOLOGY,**

**3J B INSTITUTE OF ENGINEERING AND TECHNOLOGY,**

**4J B INSTITUTE OF ENGINEERING AND TECHNOLOGY**

## **ABSTRACT**

Car parking is a major issue in modern congested cities of today. There simply are too many vehicles on the road and not enough parking spaces. We led to the need for efficient parking management systems. This we demonstrate the use of IOT based parking management system that allows for efficient parking space utilization using IOT technology. To demonstrate the concept, we use IR sensors for sensing parking slot occupancy along with a dc motor to simulate as gate opener motors. We now use a Wi-Fi modem for internet connectivity and a microcontroller for operating the system. We create a webpage for online connectivity and IOT management GUI design. The system detects the parking slots are occupied using IR sensors. The system reads the number of parking slots available or occupied and updates data with the cloud server to allow for checking parking slot availability online. This allows users to check for available parking spaces online from anywhere and available hassle-free parking. Thus, the system solves the parking users an efficient IOT based parking management system.

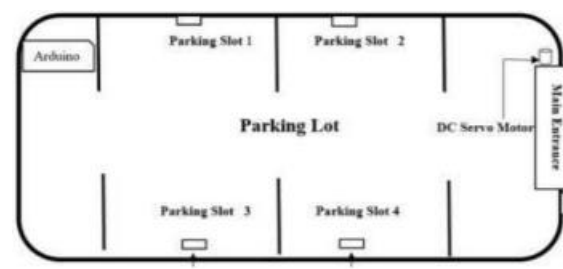
## **CHAPTER 1**

### **INTRODUCTION**

The project entitled smart parking system is to manage all the parking facilities to an user. The recent growth in economy and due to the availability of low-price cars in the market, an every average middle-class individual can afford a car, which is good thing, however the consequences of heavy traffic jams, pollution, less availability of roads and spot to drive the motor car. One of the important concerns, which is to be taken in accounting, is the problem of parking those vehicles. Though, if there is space for parking the vehicle but so much time is squandered in finding that exact parking slot resulting in more fuel intake and not also environment friendly. It will be a great deal if in some way we find out that the parking itself can provide the precise vacant position of a parking slot then it'll be helpful not limited to the drivers also for the environment. Initially when the user is about to enter the location the LCD displays the number of empty and filled spots and when the user is with its vehicle near to the parking detect sensor, he/she would be thrown with a notification on their mobile app of the parking slot number, where they should park there vehicle.

#### **1.1 Relevance of the project**

The main important benefit of a smart parking system is its advanced technology. It follows the latest technologies and concepts to assure profitable outcomes. The design and implementation of smart parking is very easy to supervise and manage. This system can be easily handled by the staff members because of its well-organized structure.



**Figure 1.1 shows the block diagram of smart parking system**

#### **1.2 Problem statement**

In recent research in metropolitan cities the parking management problem can be viewed from various angles such as high vehicle density on roads. This results in annoying issues for the drivers to park their vehicles as it is very difficult to find a parking slot. The drivers usually waste time and effort in finding parking space and end up parking their vehicles finding a space on the street which further leads to space congestion. In worst case, people fail to find any parking space especially during peak hours and festive season.

### 1.3 Objective

Smart Parking involves the use of low-cost sensors, real-time data and applications that allow users to monitor available and unavailable parking spots. The goal is to automate and decrease time spent manually searching for the optimal parking floor, spot and even lot. Some solutions will encompass a complete suite of services such as online payments, parking time notifications and even car searching functionalities for very large lots. A parking solution can greatly benefit both the user and the lot owner.

**Optimized parking** – Users find the best spot available, saving time, resources and effort. The parking lot fills up efficiently and space can be utilized properly by commercial and corporate entities.

**Reduced traffic** – Traffic flow increases as fewer cars are required to drive around in search of an open parking space.

**Reduced pollution** – Searching for parking burns around one million barrels of oil a day. An optimal parking solution will significantly decrease driving time, thus lowering the amount of daily vehicle emissions and ultimately reducing the global environmental footprint.

**Increased Safety** – Parking lot employees and security guards contain real-time lot data that can help prevent parking violations and suspicious activity. License plate recognition cameras can gather pertinent footage. Also, decreased spot-searching traffic on the streets can reduce accidents caused by the distraction of searching for parking.

**Decreased Management Costs** – More automation and less manual activity saves on labour cost and resource exhaustion.

**Enhanced User Experience** – A smart parking solution will integrate the entire user experience into a unified action. Driver's payment, spot identification, location search and time notifications all seamlessly become part of the destination arrival process.

### 1.4 Scope of the Project

At present some countries have portals which users can gain information about parking areas via the internet. This system can give users the information about parking space, but it won't be able to give which parking slot is vacant and occupied. Hence, such a system cannot smartly handle the issue. Car lifts along with an automated robotic system, which automatically takes the car to a particular parking spot as soon as the car enters on a platform. This system cannot be installed by medium scale shopping malls, movie theatres as it can cost them a huge amount. At many public places, the system only shows the availability but it cannot show the exact slot and path to the slot available. Hence, there is the need to smartly find the path to the vacant spot.

## 1.5 Methodology

In this project we are using NodeMCU, IR sensors, and servo motors. One IR sensor is used at entry and exit gate to detect the car while two IR sensors are used to detect the parking slot availability. Servo motors are used to open and close the gates according to the sensor value. NodeMCU is an open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware, which is based on the ESP-12 module. The term “NodeMCU” by default refers to the firmware rather than the dev kits. The firmware uses the Lua scripting language. The ESP8266 is a low-cost Wi-Fi enabled microchip with full TCP/IP stack and microcontroller capability. NodeMCU includes CPU core, faster Wi-Fi, more GPIOs, and supports Bluetooth 4.2, and low power Bluetooth. The ESP8266 is a low-cost Wi-Fi enabled microchip with full TCP/IP stack and microcontroller capability. NodeMCU includes CPU core, faster Wi-Fi, more GPIOs, and supports Bluetooth 4.2, and low power Bluetooth. As soon as the IR sensors get the presence of a car in front of the entrance, it will send signal to the NodeMCU to check if there is an empty slot inside the parking lot. When NodeMCU acknowledges that there is an empty slot or more then it will send a signal to the dc servo motor which will open the main entrance. On the other hand, if an NodeMCU encounters no empty slots at the time of a car trying to make an entrance, the gate will just not open. In addition, there will be a website linked with the NodeMCU board to show the number of parking.

The idea behind our methodology is very simple, usually users spend most of their time in looking for an empty slot where they can park their vehicle which increases fuel consumption and time wastage. We came-up with a new method where we provide the user an empty slot number where he can park his vehicle without wasting his time for finding one. Similarly, we try to display the start time and end time so that the user can know for what amount of time he has parked his vehicle

## CHAPTER 2

### LITERATURE SURVEY

#### **[1] Developing a Smart Parking Management System Using the Internet of Things**

Searching for parking wastes significant amounts of time and effort and leads to substantial financial costs. This is particularly the case for people who are always pressured to be on time. Smart cities employ all kinds of modern technologies to manage and enhance resources effectively. Urban parking facilities are one of the essential assets that must be managed. We developed a smart parking management system (SPMS) as a modern solution to manage parking and save users time, effort and cost. In the context of today’s modern life, it has become necessary to improve search methods for available parking and minimize the congestion that occurs at the parking entrance. Searching or booking available parking online earlier is a better substitute than searching at a parking lot where there is a possibility of not being able to find parking. Our smart parking management system was developed to:

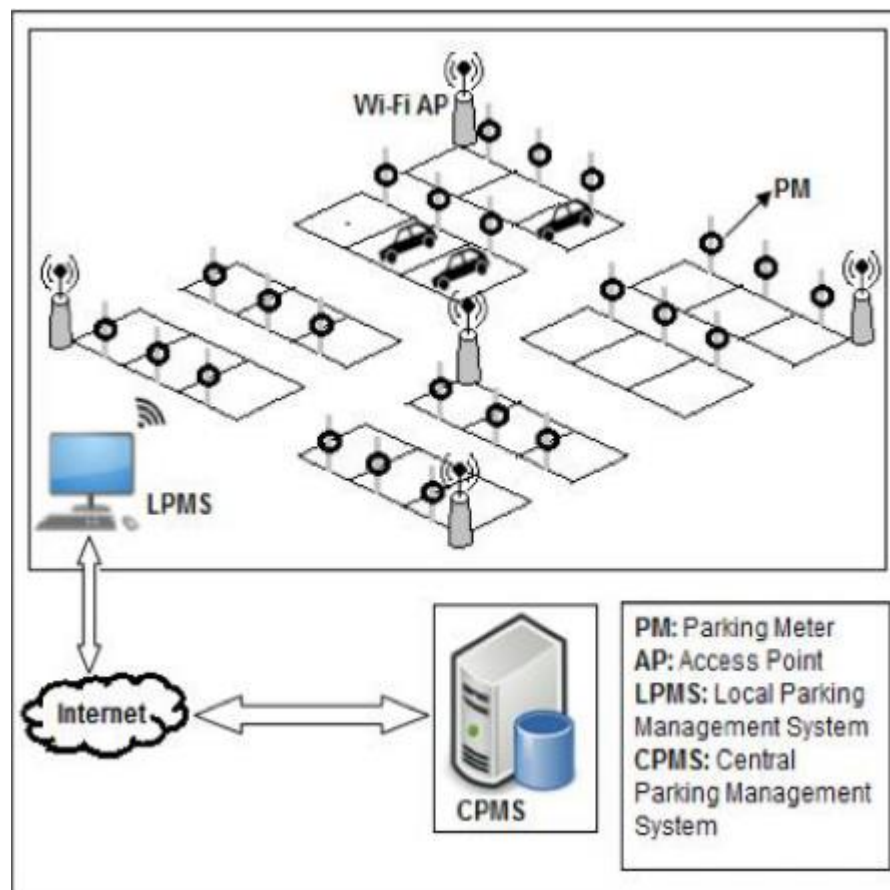


- Inquiry on availability of parking space and reservation of parking lot
- Real-time parking navigation and route guidance
- Vehicle occupancy detection and management of parking lots.

Most of the smart parking systems (SPS) proposed in literature over the past few years provides solution to the design of parking availability information system, parking reservation system, occupancy detection and management of parking lot, real-time navigation within the parking facility etc. However, very few works have paid attention to the real time detection of improper parking and automatic collection of parking charges. Thus, this paper presents an internet-of thing (IoT) based E-parking system that employs an integrated component called parking meter (PM) to address the following issues.

- Real-time detection of improper parking
- Estimation of each vehicles duration of parking lot usage
- Automatic collection of parking charges

The E-parking system proposed in this paper also provides city-wide smart parking management solution via providing parking facility availability information and parking lot reservation system and it is named as parking meter (PM) based E-parking (PM-EP).



**Fig 2.2 Network Architecture of proposed System**



### [3] Smart Parking based System for smarter cities

India is getting motorized i.e., the rate of private vehicles is more as compared to public transports. As the rate of people owning their vehicles increases, the need of parking slots to park vehicles also increases. But currently the scenario is that there are not sufficient parking slots available or there is also possibility that people are not now aware about the legal parking slots available in their locality. This situation leads to the unnecessary crowding of vehicles on the road and also results in inconveniency of people walking on the road. To overcome above problems, we are proposing the solution in the form of a multilingual android application which will be helpful for the people to find their parking slots digitally. By digitally we mean that this particular system will assign the parking slot based on the current location of the user and the parking slot which the user wants according to his/her ease. Ease in terms of finding the exact slot. The payments can be done digitally or through vending machines. The end user can register and login with his/her account which will help the system to find the location and displaying the nearest parking area and nearest parking slot, whether it is available or not. If not then it will direct user to the next nearest slot and so on. The existing system comprises of both traditional and application-based approach for parking. If we talk about the traditional approach, it utilizes manual method of parking i.e user has to find the spot for parking by traveling to far distances and paying extra money. An application-based approach consists of the applications which provides the parking slots for the particular locality for example. The application named '**Parking Panda**' provides the parking slots to the areas like stadium, sports leagues etc.

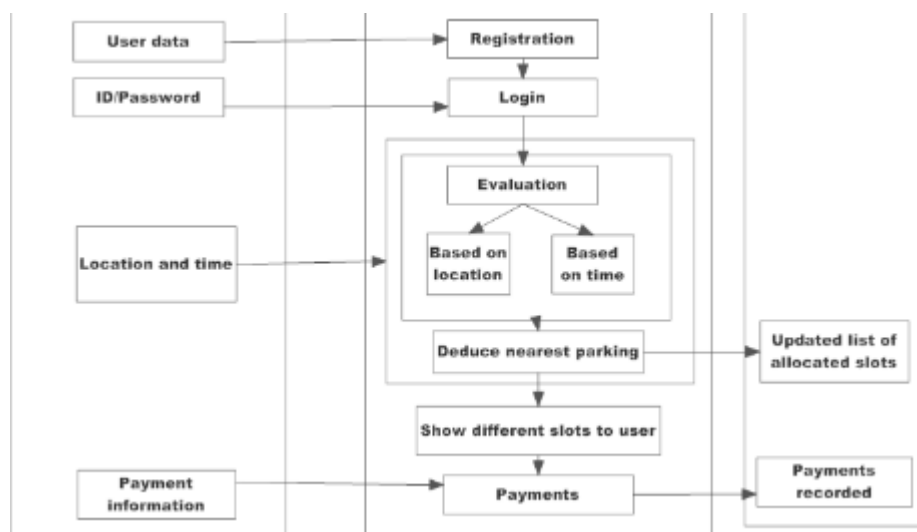


Fig 2.3 Block Diagram of parking system

### [4] SMART PARKING SYSTEM TO REDUCE TRAFFIC CONGESTION

Transportation is the key-success for any of the country. Now a day, many people have options to use their own vehicle for travelling. This will surely increase the demand in trading but one of the problems created by road traffic is "parking". To park all these vehicles in the major metro cities is quite tedious and difficult task and it became problematic to park vehicles. Lot of research and development is being done all over the world to implement better and smarter parking management mechanisms. The current smart parking systems or Wireless Sensors Network Parking requires the combination of wireless sensor networks module, Embedded web-server, Central Web- Server. Sensor networks make use of Infrared (IR) Sensor nodes to check the parking slot state and send this

information to embedded web-server. It thereby displays the information on a LED screen with which the user can check for empty vehicle slots. These systems not guide the users to reach to the parking lot. If the slot is not available at that time than drivers will start searching for another parking zone so that this process is time consuming and will increase the traffic congestion. This paper proposes a Reservation-based Smart Parking System for avoiding the traffic problems that provides the pre- booking of slots through the use of the mobile application. This application is expected to provide an efficient and cost- effective solution to the vehicle parking problems. Application must be installed in the user's mobile. Unlike the existing system, our idea is to use client-server architecture where client request for the reservation of slots and server responds with the slots which are available at that time. Our system is that the user has an option to go for the parking area according to his/her convenience. The advantage of this will greatly reduce the time taken by the vehicle to search for a parking area. Advanced payment modules are also included like e- wallet, debit card, credit card from which the user can pay. Penalty will be added on late exit as well as an over use of the slot after user specified entry and exit time. The refund will be given on cancelation of parking slot and early exit. The supervisor is required to monitor the area. Many of the vehicles parking facilities are unable to cope with the influx of vehicles on roads and parking area. The current smart parking systems or Wireless Sensors Network Parking requires the combination of wireless sensor networks module, Embedded web-server, Central Web-Server. Sensor networks make use of Infrared (IR) Sensor nodes to check the parking slot state and send this information to embedded web-server. It thereby displays the information on a LED screen with which the user can check for empty vehicle slots. Also, image capturing devices are used for continuously clicking pictures of parking area to ensure empty slots which results in high power consumption and also high maintenance cost is required. There are some systems in the market like the smart parking services which are based on the wireless sensor networks which uses wireless sensors to effectively find the available parking space. But to use this system, additional hardware needs to be installed in the car which is not feasible. Finding a parking slot in a congested city is very hard. In many cases people go to a parking station and they find it full and there is no space available for parking. Then in search of parking space they have to again roam with their vehicle to find available parking.



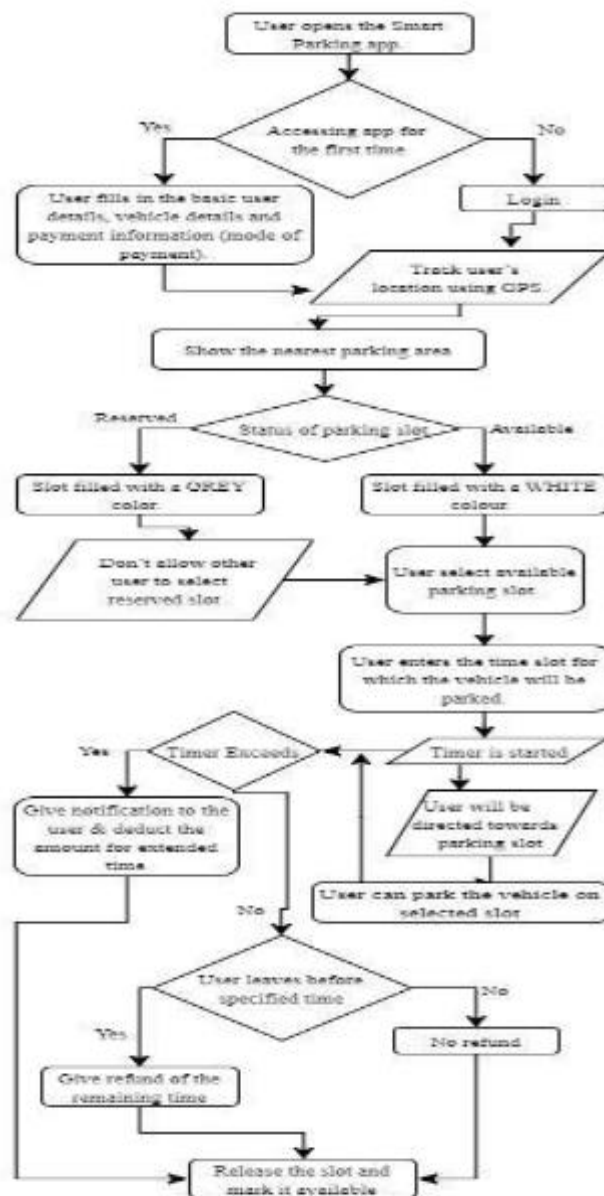
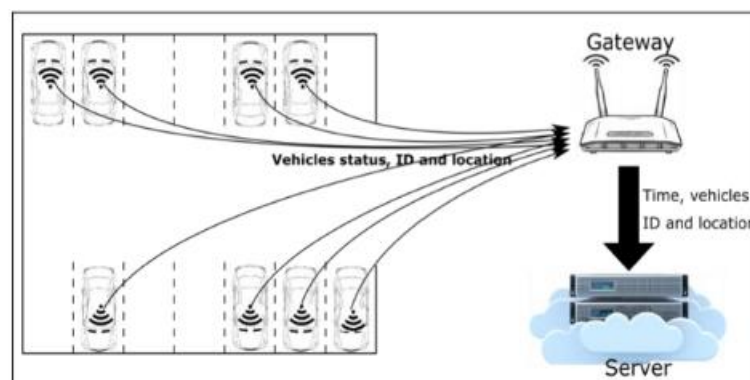


Fig 2.4 Flow chart of Smart Parking System

### **[5] An IoT-Based Intelligent System for Real-Time Parking Monitoring and Automatic Billing**

Today, the parking industry is being transformed by new technologies that are allowing cities to reduce rates of congestion significantly. Sensor networks that sense vehicle occupancy are providing the basic intelligence behind smart parking systems. Thanks to the Smart Parking technology, it is now possible to know in real-time the location of free parking spaces and to help drivers to get to their ultimate destination. A variety type of vehicle detectors has been used in parking information acquisition. These vehicle detectors mainly include the inductive loop, acoustic sensor, infrared sensor, or ultrasonic sensor. System using video camera sensor technologies have been proposed to collect the information in vehicle parking field. However, a video camera sensor is vulnerable to bad weather and night time operation. Furthermore, it is expensive, and can generate a large amount of data that can be difficult to transmit in a wireless network. The magneto-resistive based detection systems combined with a wireless area network are the most popular technique due to their high accuracy. Yet, this type of sensor is facing different issues, i.e., it can be bedevilled by electromagnetic interference, which affects the accuracy, the reading from sensor needs to be collected constantly which will result in wearing out the battery. To extend the battery lifetime and increase the vehicle detection accuracy, a parking sensor system has been proposed. While power management technique has been implemented to

optimize energy consumption, high occupancy monitoring accuracy is achieved using two-fold sensing approach. It is a sequence of darkness and Signal Strength Indicator (RSSI) measurement-based techniques. The wireless sensors are still intrusive, they are embedded in the pavement, or taped to the surface of each individual parking lot. Existing sensors, such as ground based parking sensors costs up to \$200 per parking lot. As consequence, smart-parking technology using wireless sensors for outdoor parking is costly due to the large number of sensors units required to cover the entire parking lot. Although, parking occupancy monitoring systems have made a significant progress, smart parking payment is rarely studied in smart parking research. Yet, there are companies working on the patents of parking systems for payments. A first approach consists in using a camera or an RFID transceiver for vehicle detection and identification. A limitation of this solution lies in that the system is complex and its implementation is expensive when a detection device is installed on each parking lot. Furthermore, when only RFID transceiver is used for vehicle detection and identification, the system can be bedevilled by electromagnetic interference, which affects the accuracy. Moreover, this system is designed to detect a vehicle when entering a parking and seek payment, whereas information on vacant parking lots is not provided. A technique for monitoring vehicle parking using one camera to record the entrance of a vehicle and a second camera to record the vehicle leaving the parking has been proposed. Moreover, in a system and method for obtaining and displaying information on vacant parking space is described. When a user occupies a parking space designated with an individual ID, he enters this ID into a parking meter or via a smart phone mobile app., and pays the parking fees. The database processes the received data and changes the status of the parking space with its ID from unpaid to paid. These data are used as information on the occupation of a parking space. In this paper, we propose a smart sensor system allowing outdoor parking monitoring and payment without requiring any user/driver interaction. It will be deployed without having to install new components on each parking lot. The proposed sensor has benefits in terms of detection and payment reliability, and reduced expense by reducing the system complexity and installation, and extending batteries lifetime through the reduction of the system power consumption.



**Fig. 2.5. Proposed system architecture; wireless occupancy sensor; wireless gateway; data storage and processing unit.**

**Table 2.1 Summary of the Approaches**





| <b><u>Approaches</u></b>   |  |
|--|--|
| It deals with saving financial cost by developing the system in the most efficient manner. | Keeps a count on number of vacant spots and prior booking. |
| Real time monitoring vehicle occupancy   | Displays appropriate message                               |
| Android app for providing all parking  | Specifying every minute                                    |
| Details to the user.   | Information to the user using android application.         |

## **CHAPTER- 3**

### **SYSTEM REQUIRMENTS SPECIFICATION**

#### **3.1 FUNCTIONAL REOURIMENTS**




Functional Requirement defines a function of a software system and how the system must behave when presented with specific inputs/or conditions. These may include calculations, data manipulation and processing and other specific functionality. In these systems following are the functional requirements

-  The application should not display in-appropriate message for valid conditions.
-  The application must not stop working when kept running for even a long time.
-  The application should process information for any kind of input case.
-  The application should generate the output for a given input test case.

#### **3.2 NON-FUNCTIONAL REQUERMENTS**

Non-functional requirements are the requirements which are not directly concerned with the specific function delivered by the system. They specify the criteria that can be used to judge the operation of a system rather than specific behaviours.

Given below are the non-functional requirements:

-  Product requirements
-  Organizational requirements
-  Basic operational requirements

### 3.3 HARDWARE SPECIFICATIONS

- ✚ NODEMCU
- ✚ ARDUINO UNO
- ✚ IR SENSORS
- ✚ LED LIGHTS
- ✚ SERVO MOTORS
- ✚ LCD DISPLAY

### 3.4 SOFTWARE DESCRIPTION

- ✚ ARDUINO IDE

## CHAPTER 4

### SYSTEM ANALYSIS AND DESIGN

#### 4.1 NODEMCU

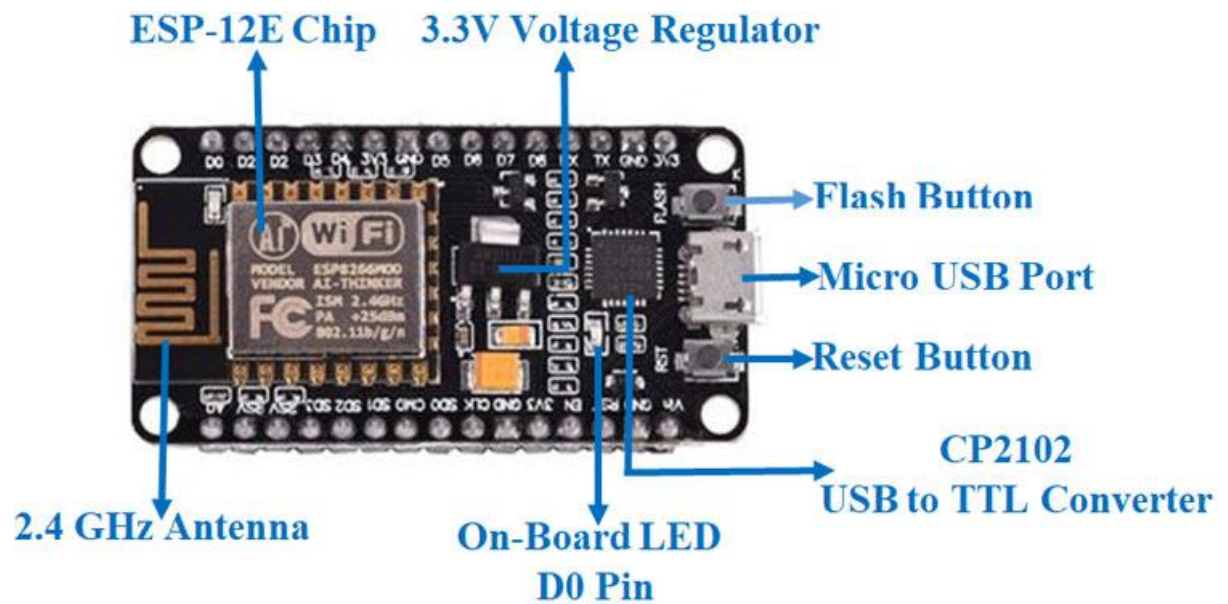
NodeMCU is an open-source Lua based firmware and **development board** specially targeted for IoT based Applications. It includes firmware that runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module.

#### NodeMCU Specification & Features

- ✚ Microcontroller: Tensilica 32-bit RISC CPU Xtensa LX106
- ✚ Operating Voltage: 3.3V
- ✚ Input Voltage: 7-12V
- ✚ Digital I/O Pins (DIO): 16
- ✚ Analog Input Pins (ADC): 1
- ✚ UARTs: 1
- ✚ SPIs: 1
- ✚ I2Cs: 1
- ✚ Flash Memory: 4 MB
- ✚ SRAM: 64 KB
- ✚ Clock Speed: 80 MHz
- ✚ USB-TTL based on CP2102 is included onboard, Enabling Plug n Play
- ✚ PCB Antenna
- ✚ Small Sized module to fit smartly inside your IoT projects

The **NodeMCU ESP8266 development board** comes with the ESP-12E module containing the ESP8266 chip having Tensilica Xtensa 32-bit LX106 RISC microprocessor. This microprocessor supports RTOS and operates at 80MHz to 160 MHz adjustable clock frequency. NodeMCU has 128 KB RAM and 4MB of Flash memory to store data and programs. Its high processing power with in-built Wi-Fi / Bluetooth and Deep Sleep Operating features make it ideal for IoT projects.

NodeMCU can be powered using a Micro USB jack and VIN pin (External Supply Pin). It supports UART, SPI, and I2C interface.



## Applications

- ✚ Prototyping of IoT devices
- ✚ Low power battery operated applications
- ✚ Networking Projects
- ✚ Projects requiring multiple I/O interfaces with Wi-Fi and Bluetooth functionalities.

## NodeMCU Pin configuration

| Pin Category | Name                            | Description   |
|--------------|---------------------------------|---|
| Power        | Micro-USB,<br>3.3V, GND,<br>Vin | <b>Micro-USB:</b> NodeMCU can be powered through the USB port<br><br><b>3.3V:</b> Regulated 3.3V can be supplied to this pin to power the board<br><br><b>GND:</b> Ground pins<br><br><b>Vin:</b> External Power Supply |
| Control Pins | EN, RST                         | The pin and the button resets the microcontroller   |
| Analog Pin   | A0                              | Used to measure analog voltage in the range of 0-3.3V   |

|           |                        |   |
|-----------|------------------------|---|
| GPIO Pins | GPIO1 to GPIO16        | NodeMCU has 16 general purpose input-output pins on its board   |
| SPI Pins  | SD1, CMD, SD0, CLK     | NodeMCU has four pins available for SPI communication.  |
| UART Pins | TXD0, RXD0, TXD2, RXD2 | NodeMCU has two UART interfaces, UART0 (RXD0 & TXD0) and UART1 (RXD1 & TXD1). UART1 is used to upload the firmware/program.   |
| I2C Pins  |                        | NodeMCU has I2C functionality support but due to the internal functionality of these pins, you have to find which pin is I2C. |

## 4.2 ARDUINO UNO

Arduino UNO is a low-cost, flexible, and easy-to-use programmable open-source microcontroller board that can be integrated into a variety of electronic projects. This board can be interfaced with other Arduino boards, Arduino shields, Raspberry Pi boards and can control relays, LEDs, servos, and motors as an output.




Arduino UNO features AVR microcontroller Atmega328, 6 analogue input pins, and 14 digital I/O pins out of which 6 are used as PWM output.

This board contains a USB interface i.e. USB cable is used to connect the board with the computer and Arduino IDE (Integrated Development Environment) software is used to program the board.

The unit comes with 32KB flash memory that is used to store the number of instructions while the SRAM is 2KB and EEPROM is 1KB.

The operating voltage of the unit is 5V which projects the microcontroller on the board and its associated circuitry operates at 5V while the input voltage ranges between 6V to 20V and the recommended input voltage ranges from 7V to 12V.

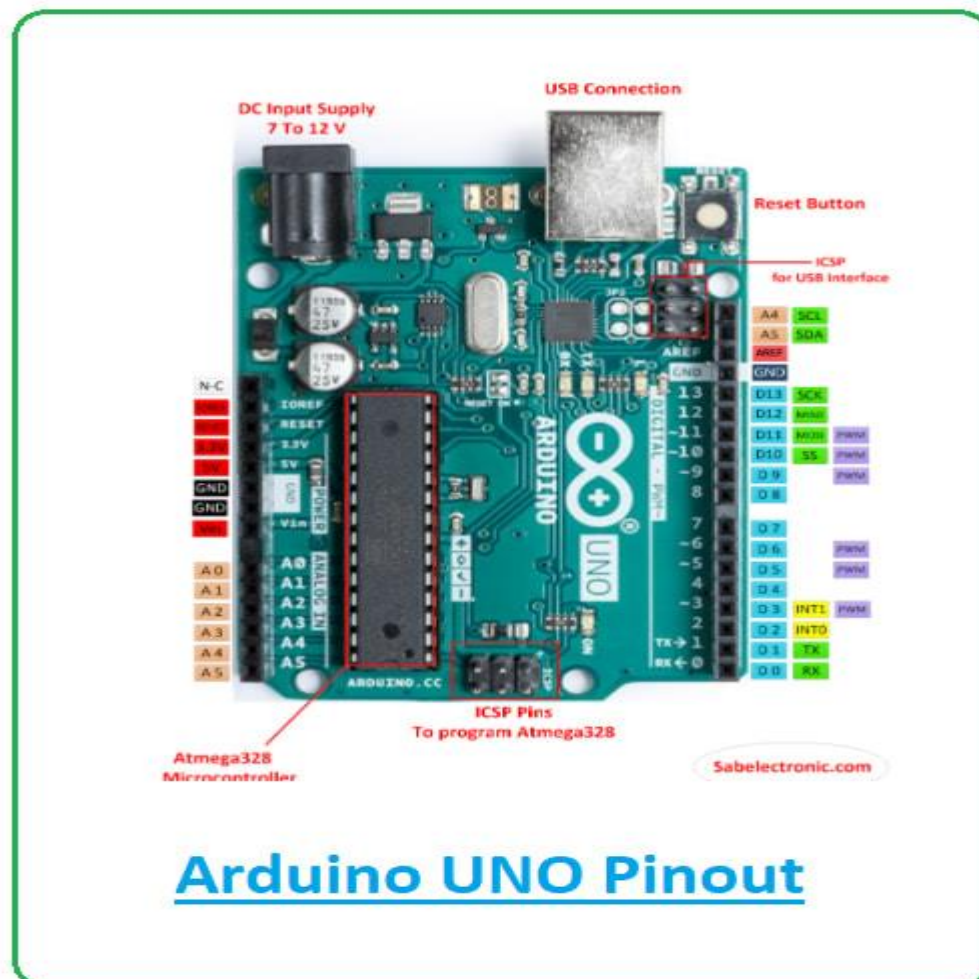
## ARDUINO UNO COMPONENTS

-  **ATmega328:** This is the brain of the board in which the program is stored.
-  **PWM:** the board contains 6 PWM pins. PWM stands for Pulse Width Modulation, using this process we can control the speed of the servo motor, DC motor, and brightness of the LED.
-  **Digital I/O Pins:** there are 14 digital (0-13) I/O pins available on the board that can be connected with external electronic components



- ✚ **Analogue Pins:** there are 6 analogue pins integrated on the board. These pins can read the analogue sensor and can convert it into a digital signal.
- ✚ **AREF:** It is an Analog Reference Pin used to set an external reference voltage.
- ✚ **Reset Button:** This button will reset the code loaded into the board. This button is useful when the board hangs up, pressing this button will take the entire board into an initial state.
- ✚ **USB Interface:** This interface is used to connect the board with the computer and to upload the Arduino sketches
- ✚ **DC Power Jack:** This is used to power up the board with a power supply
- ✚ **Power LED:** This is a power LED that lights up when the board is connected with the power source.
- ✚ **Micro SD Card:** The UNO board supports a micro SD card that allows the board to store more information
- ✚ **3.3V:** This pin is used to supply 3.3V power to your projects
- ✚ **5V:** This pin is used to supply 5V power to your projects.
- ✚ **VIN:** It is the input voltage applied to the UNO board
- ✚ **Voltage Regulator:** The voltage regulator controls the voltage that goes into the board.
- ✚ **SPI:** The SPI stands for Serial Peripheral Interface. Four Pins 10(SS), 11(MOSI), 12(MISO), 13(SCK) are used for this communication.
- ✚ **TX/RX:** Pins TX and RX are used for serial communication. The TX is a transmit pin used to transmit the serial data while RX is a receive pin used to receive serial data.

## ARDUINO UNO Pinout



An infrared (IR) sensor is an electronic device that measures and detects infrared radiation in its surrounding environment. Infrared radiation was accidentally discovered by an astronomer named William Herchel in 1800. While measuring the temperature of each colour of light (separated by a prism), he noticed that the temperature just beyond the red light was highest. IR is invisible to the human eye, as its wavelength is longer than that of visible light (though it is still on the same electromagnetic spectrum). Anything that emits heat (everything that has a temperature above around five degrees Kelvin) gives off infrared radiation. We are using three IR detect sensor in our project, one IR detect sensor is used to sense the vehicle near the parking sensor and other two IR detect sensor is used to send data to the Node MCU which is the brain of our system whether a vehicle is parked in that slot or is unparked

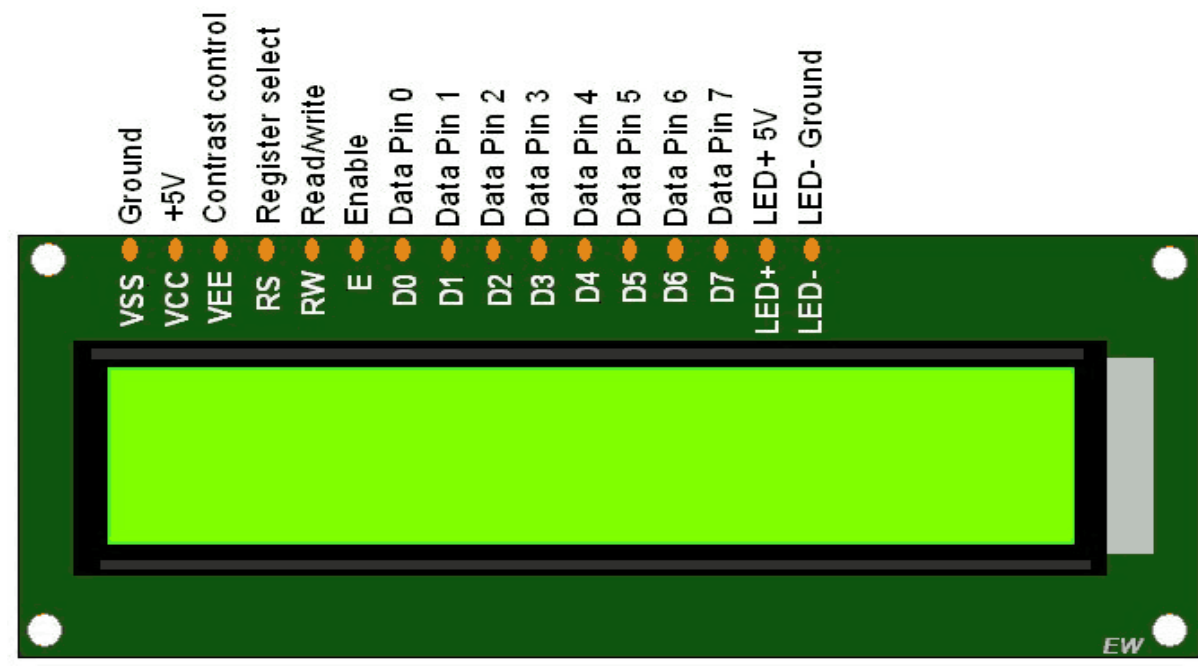
There are two types of infrared sensors: active and passive. Active infrared sensors both emit and detect infrared radiation. Active IR sensors have two parts: a light emitting diode (LED) and a receiver. When an object comes close to the sensor, the infrared light from the LED reflects off of the object and is detected by the receiver. Active IR sensors act as proximity sensors, and they are commonly used in obstacle detection systems (such as in robots).

- ✚ **Active output level:** Outputs low logic level when obstacle is detected.
- ✚ On board obstacles detection LED indicator
- ✚ Ultrasonic sensor is used in this parking system
- ✚ The range of the IR sensor is between 1-2 meters/3-15 feet.

An LCD is an electronic display module which uses liquid crystal to produce a visible image. The 16×2 LCD display is a very basic module commonly used in DIY's and circuits. The 16×2 translates to a display 16 characters per line in 2 such lines. In this LCD each character is displayed in a 5×7-pixel matrix. The 16\*2 display is used to display the number of vacant and spilled spot. It also gets updated on the display LCD when a vehicle parks or unparks the vehicle.

## TYPES OF LCDs

- ✚ **Twisted Nematic (TN)**- which are inexpensive while having high response times. However, TN displays have low contrast ratios, viewing angles and colour contrasts.
- ✚ **In Panel Switching displays (IPS Panels)**- which boast much better contrast ratios, viewing angles and colour contrast when compared to TN LCDs.
- ✚ **Vertical Alignment Panels (VA Panels)**- which are seen as a medium quality between TN and IPS displays.
- ✚ **Advanced Fringe Field Switching (AFFS)**- which is a top performer compared IPS displays in colour reproduction range.



## SERVO MOTORS

A servomotor (or servo motor) is **a rotary actuator or linear actuator that allows for precise control of angular or linear position, velocity and acceleration**. It consists of a suitable motor coupled to a sensor for position feedback.

In this project we use servo motors to open the gate of the parking slot when the vehicle is detected at the entrance.

When the car is leaving the parking slots the servo motors helps to lift the gate and helps the car to leave the parking slots.



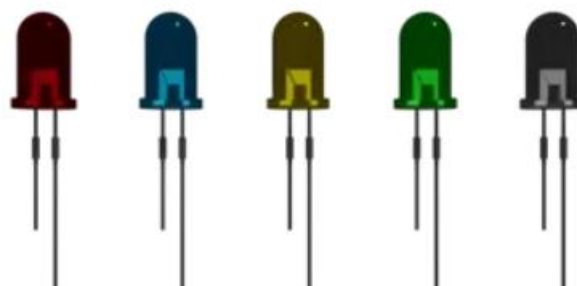
## LED [LIGHT EMITTING DIODE]

A **Light Emitting Diode (LED)** is a special type of PN junction diode. The light emitting diode is specially doped and made of a special type of semiconductor. This diode can emit light when it is in the forward biased state. Aluminium indium gallium phosphide (AlInGaP) and indium gallium nitride (InGaN) are two of the most commonly used semiconductors for LED technologies

## **Colour of an LED**

The colour of an LED device is expressed in terms of the dominant wavelength emitted,  $\lambda_d$  (in nm). AlInGaP LEDs produce the colours red (626 to 630 nm), red-orange (615 to 621 nm), orange (605 nm), and amber (590 to 592 nm). InGaN LEDs produce the colours green (525 nm), blue green (498 to 505 nm), and blue (470 nm). The colour and forward voltage of AlInGaP LEDs depend on the temperature of the LED p-n junction.

As the temperature of the LED p-n junction increases, the luminous intensity decreases, the dominant wavelength shifts towards longer wavelengths, and the forward voltage drops. The variation in luminous intensity of InGaN LEDs with operating ambient temperature is small (about 10%) from  $-20^{\circ}\text{C}$  to  $80^{\circ}\text{C}$ . However, the dominant wavelength of InGaN LEDs does vary with LED drive current; as the LED drive current increases, dominant wavelength moves toward shorter wavelengths.



In this project LED lights are used to indicate whether the car is present in the parking slot or not. In this smart parking system, we use RED, GREEN color LED lights. The RED color is used to indicate the parking slot is full. GREEN LED Light is used to indicate that the parking slot is vacant and there is a space for parking car in the parking slot.

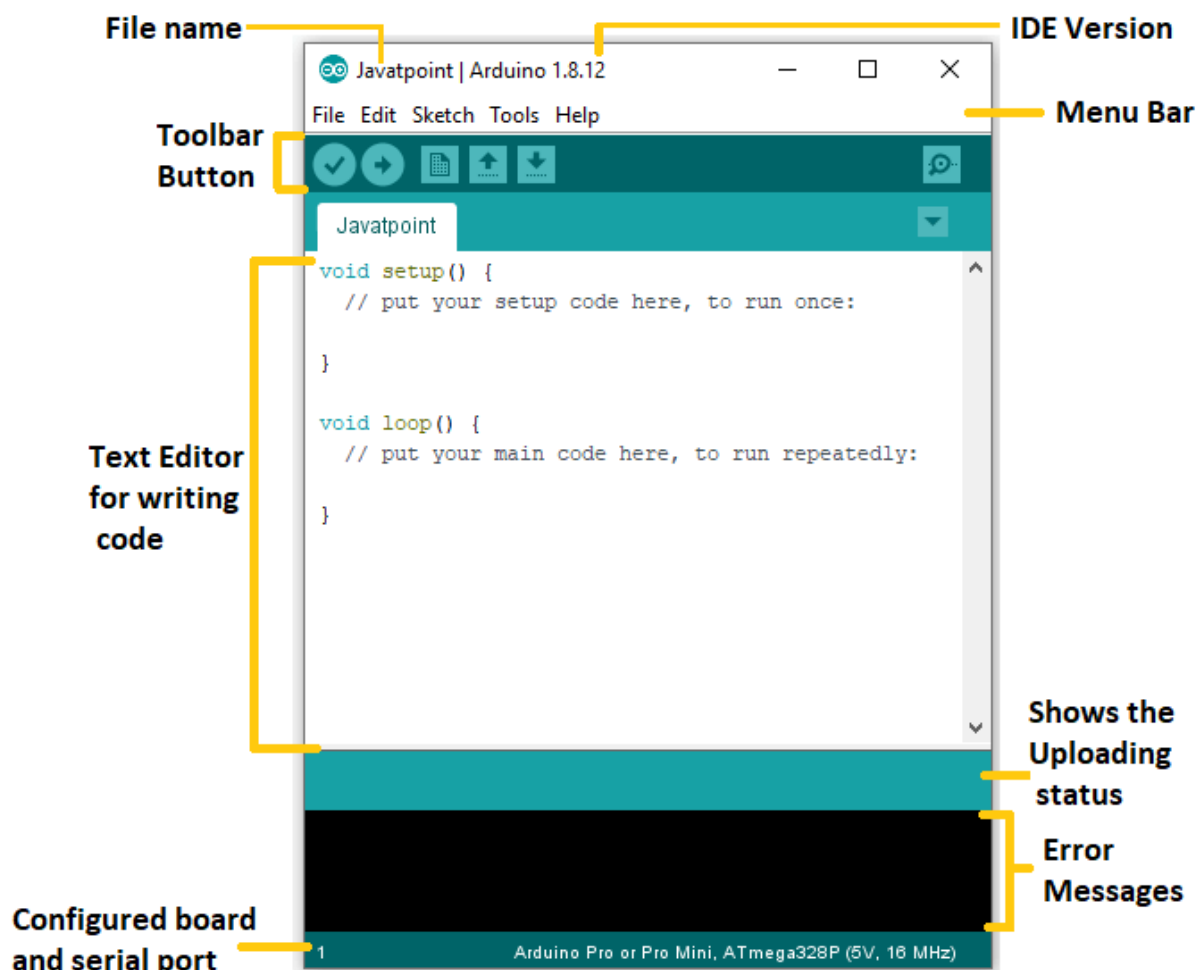


## Arduino IDE

The Arduino IDE is an open-source software, which is used to write and upload code to the Arduino boards. The IDE application is suitable for different operating systems such as **Windows, Mac OS X, and Linux**. It supports the programming languages C and C++. Here, IDE stands for **Integrated Development Environment**.

The program or code written in the Arduino IDE is often called as sketching. We need to connect the Genuino and Arduino board with the IDE to upload the sketch written in the Arduino IDE software. The sketch is saved with the extension '.ino.'

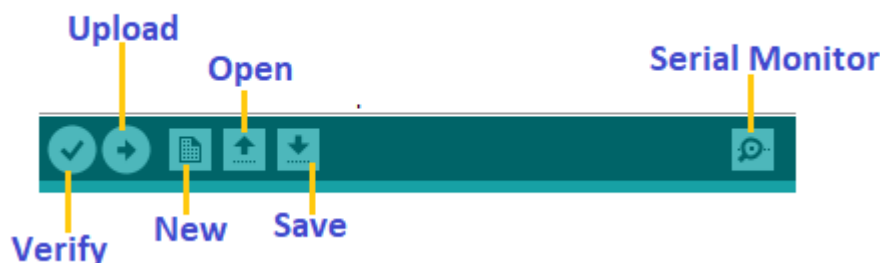
## The Arduino IDE will appear as



## Toolbar Button

The icons displayed on the toolbar are **New, Open, Save, Upload, and Verify**.

It is shown below:



## Upload

The Upload button compiles and runs our code written on the screen. It further uploads the code to the connected board. Before uploading the sketch, we need to make sure that the correct board and ports are selected.

We also need a USB connection to connect the board and the computer. Once all the above measures are done, click on the Upload button present on the toolbar.

The latest Arduino boards can be reset automatically before beginning with Upload. In the older boards, we need to press the Reset button present on it. As soon as the uploading is done successfully, we can notice the blink of the Tx and Rx LED.

If the uploading is failed, it will display the message in the error window.

We do not require any additional hardware to upload our sketch using the Arduino Bootloader. A **Bootloader** is defined as a small program, which is loaded in the microcontroller present on the board. The LED will blink on PIN 13.

## Open

The Open button is used to open the already created file. The selected file will be opened in the current window.

## Save

The save button is used to save the current sketch or code.

## New

It is used to create a new sketch or opens a new window.

## Verify

The Verify button is used to check the compilation error of the sketch or the written code.



## Serial Monitor

The serial monitor button is present on the right corner of the toolbar. It opens the serial monitor.

It is shown below:

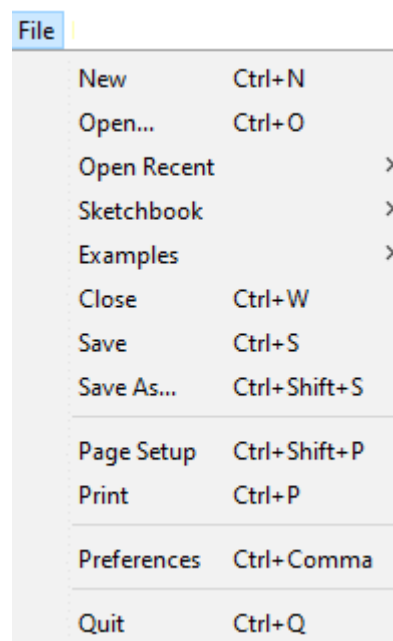
When we connect the serial monitor, the board will reset on the operating system Windows, Linux, and Mac OS X. If we want to process the control characters in our sketch, we need to use an external terminal program. The terminal program should be connected to the COM port, which will be assigned when we connect the board to the computer.

## Menu Bar



**File**

When we click on the File button on the Menu bar, a drop-down list will appear. It is shown below:



Let's discuss each option in detail.

### New

The New button opens the new window. It does not remove the sketch which is already present.

### Open

It allows opening the sketch, which can be browsed from the folders and computer drivers.

### Open Recent

The Open Recent button contains the list of the recent sketches.

## Sketchbook

It stores the current sketches created in the Arduino IDE software. It opens the selected sketch or code in a new editor at an instance.

## Examples

It shows the different examples of small projects for a better understanding of the IDE and the board. The IDE provides examples of self-practice.

## Close

The Close button closes the window from which the button is clicked.

## Save

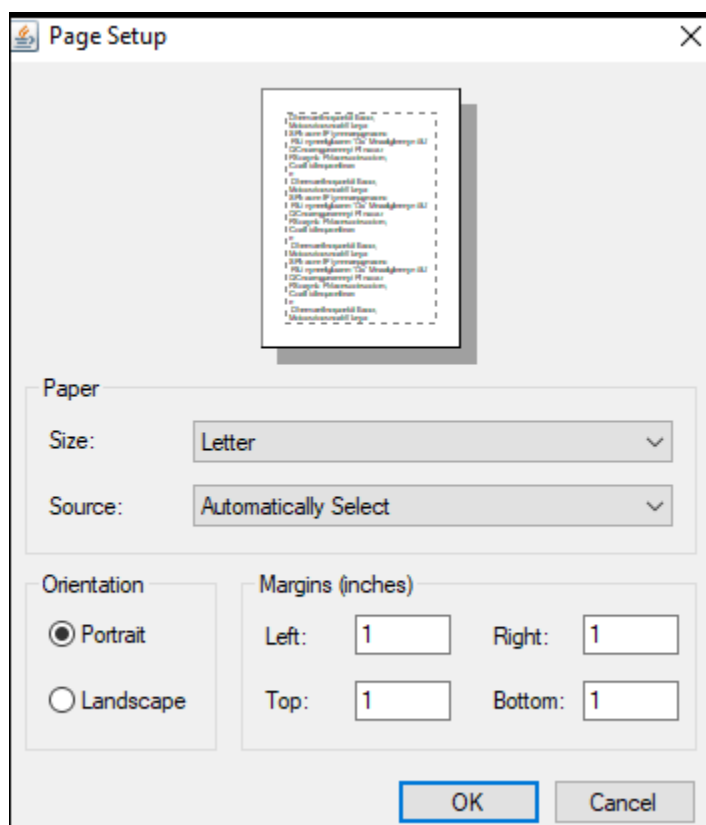
The save button is used to save the current sketch. It also saves the changes made to the current sketch. If we have not specified the name of the file, it will open the 'Save As...' window.

## Save As...

We can save the sketch with a different name using the 'Save As...' button. We can also change the name accordingly.

## Page Setup

It allows setting the page margins, orientation, and size for printing. The 'Page Setup' window will appear as:



## Print

According to the settings specified in the 'Page Setup', it prepares the current sketch for printing.

## Preferences

It allows the customization settings of the Arduino IDE.

## Quit

The Quit button is used to close all the IDE windows. The same closed sketch will be reopened when we will open the Arduino IDE.

## Edit

When we click on the Edit button on the Menu bar, a drop-down list appears. It is shown below:

| Edit               |              |
|--------------------|--------------|
| Undo               | Ctrl+Z       |
| Redo               | Ctrl+Y       |
| <hr/>              |              |
| Cut                | Ctrl+X       |
| Copy               | Ctrl+C       |
| Copy for Forum     | Ctrl+Shift+C |
| Copy as HTML       | Ctrl+Alt+C   |
| Paste              | Ctrl+V       |
| Select All         | Ctrl+A       |
| Go to line...      | Ctrl+L       |
| <hr/>              |              |
| Comment/Uncomment  | Ctrl+Slash   |
| Increase Indent    | Tab          |
| Decrease Indent    | Shift+ Tab   |
| <hr/>              |              |
| Increase Font Size | Ctrl+Plus    |
| Decrease Font Size | Ctrl+Minus   |
| <hr/>              |              |
| Find...            | Ctrl+F       |
| Find Next          | Ctrl+G       |
| Find Previous      | Ctrl+Shift+G |

Let's discuss each option in detail.

## Undo

The Undo button is used to reverse the last modification done to the sketch while editing.

## Redo

The Redo button is used to repeat the last modification done to the sketch while editing.

## Cut

It allows us to remove the selected text from the written code. The text is further placed to the clipboard. We can also paste that text anywhere in our sketch.

## Copy

It creates a duplicate copy of the selected text. The text is further placed on the clipboard.

## Copy for Forum

The 'Copy for Forum' button is used to copy the selected text to the clipboard, which is also suitable for posting to the forum.

## Copy as HTML

The 'Copy for Forum' button is used to copy the selected text as HTML to the clipboard. It is desirable for embedding in web pages.

## Paste

The Paste button is used to paste the selected text of the clipboard to the specified position of the cursor.

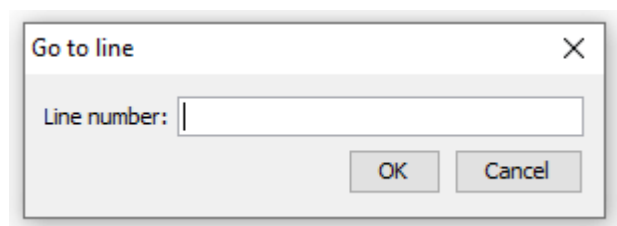
## Select All

It selects all the text of the sketch.

## Go to line...

It moves the cursor to the specified line number.

The window will appear as:



## Comment/Decomment

The Comment/ Decomment button is used to put or remove the comment mark (//) at the beginning of the specified line.

## Increase Indent

It is used to add the space at the starting of the specified line. The spacing moves the text towards the right.

## Decrease Indent

It is used to subtract or remove the space at the starting of the specified line. The spacing moves the text towards the left.

## Increase Font Size

It increases the font size of the written text.

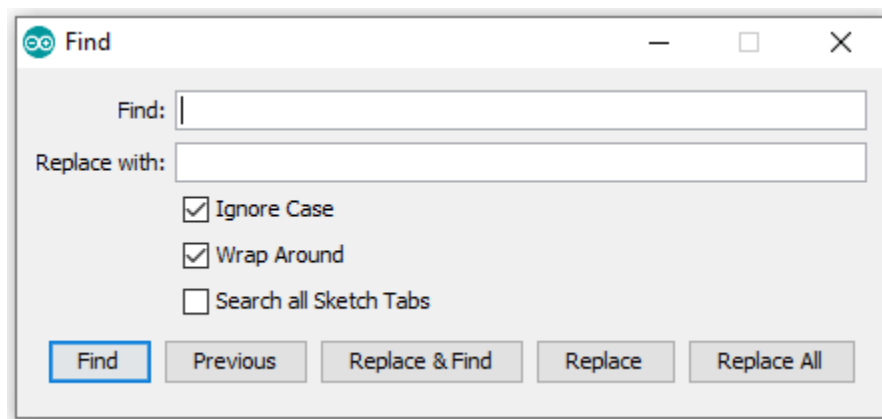
## Decrease Font Size

It decreases the font size of the written text.

## Find...

It is used to find the specified text. We can also replace the text. It highlights the text in the sketch.

The window will appear as:



## Find Next

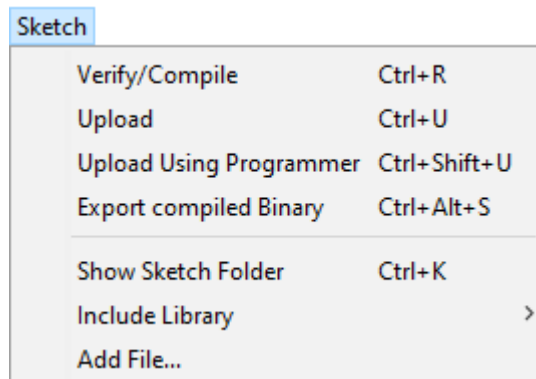
It highlights the next word, which has specified in the '**Find...**' window. If there is no such word, it will not show any highlighted text.

## Find Previous

It highlights the previous word, which has specified in the '**Find...**' window. If there is no such word, it will not show any highlighted text.

- **Sketch**

When we click on the Sketch button on the Menu bar, a drop-down list appears. It is shown below:



Let's discuss each option in detail.

### **Verify/Compile**

It will check for the errors in the code while compiling. The memory in the console area is also reported by the IDE.

### **Upload**

The Upload button is used to configure the code to the specified board through the port.

### **Upload Using Programmer**

It is used to override the Bootloader that is present on the board. We can utilize the full capacity of the Flash memory using the '**Upload Using Programmer**' option. To implement this, we need to restore the Bootloader using the **Tools-> Burn Bootloader** option to upload it to the USB serial port.

### **Export compiled Binary**

It allows saving a **.hex** file and can be kept archived. Using other tools, **.hex** file can also be sent to the board.

### **Show Sketch Folder**

It opens the folder of the current code written or sketch.

### **Include Library**

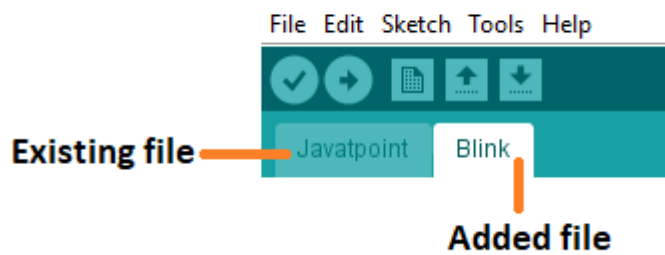
Include Library includes various Arduino libraries. The libraries are inserted into our code at the beginning of the code starting with the **#**. We can also import the libraries from **.zip** file.

### **Add File...**

The Add File... button is used to add the created file in a new tab on the existing file.

For example, let's add '**Blink**' file to the '**Javatpoint**' file. The tab will now appear as:

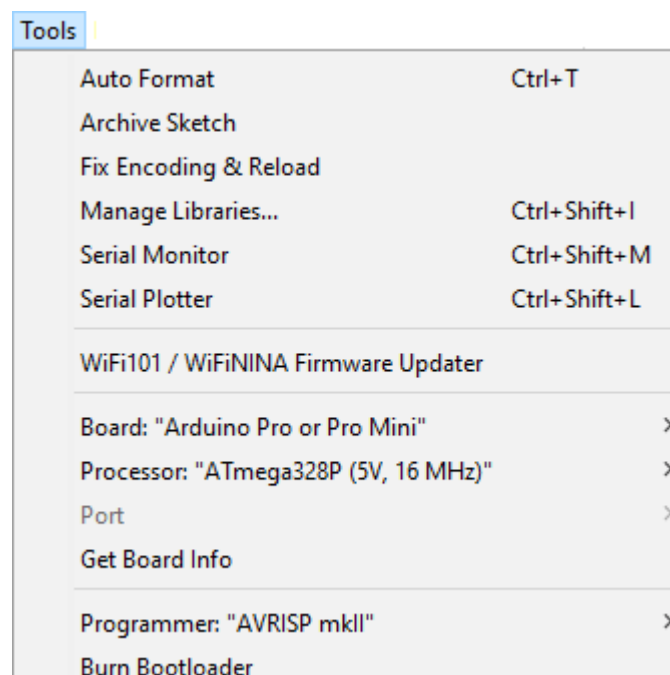




We can also delete the corresponding file from the tab by clicking on the **small triangle** -> **Delete** option.

## Tools

When we click on the Tools button on the Menu bar, a drop-down list appears. It is shown below:



Let's discuss each option in detail.

## Auto Format

The Auto Format button is used to format the written code. For example, lining the open and closed curly brackets in the code.

## Archive Sketch

The copy of the current sketch or code is archived in the .zip format. The directory of the archived is same as the sketch.

## Fix Encoding and Reload

This button is used to fix the inconsistency between the operating system char maps and editor char map encoding.

## **Manage Libraries...**

It shows the updated list of all the installed libraries. We can also use this option to install a new library into the Arduino IDE.

## **Serial Monitor**

It allows the exchange of data with the connected board on the port.

## **Serial Plotter**

The Serial Plotter button is used to display the serial data in a plot. It comes preinstalled in the Arduino IDE.

## **WiFi101/WIFI NINA Firmware Updater**

It is used to check and update the Wi-Fi Firmware of the connected board.

## **Board**

We are required to select the board from the list of boards. The selected board must be similar to the board connected to the computer.

## **Processor**

It displays the processor according to the selected board. It refreshes every time during the selection of the board.

## **Port**

It consists of the virtual and real serial devices present on our machine.

## **Get Board Info**

It gives the information about the selected board. We need to select the appropriate port before getting information about the board.

## **Programmer**

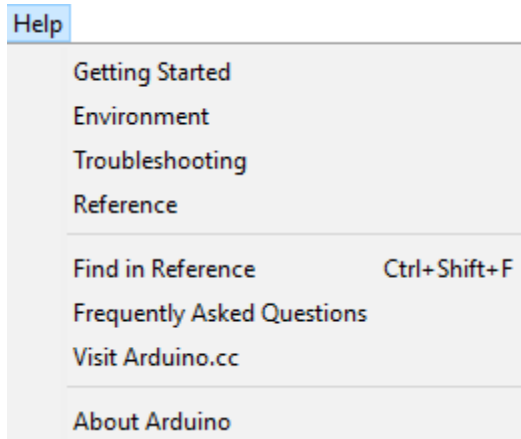
We need to select the hardware programmer while programming the board. It is required when we are not using the onboard USB serial connection. It is also required during the burning of the Bootloader.

## **Burn Bootloader**

The Bootloader is present on the board onto the microcontroller. The option is useful when we have purchased the microcontroller without the bootloader. Before burning the bootloader, we need to make sure about the correct selected board and port.

## **Help**

When we click on the Help button on the Menu bar, a drop-down list will appear. It is shown below:

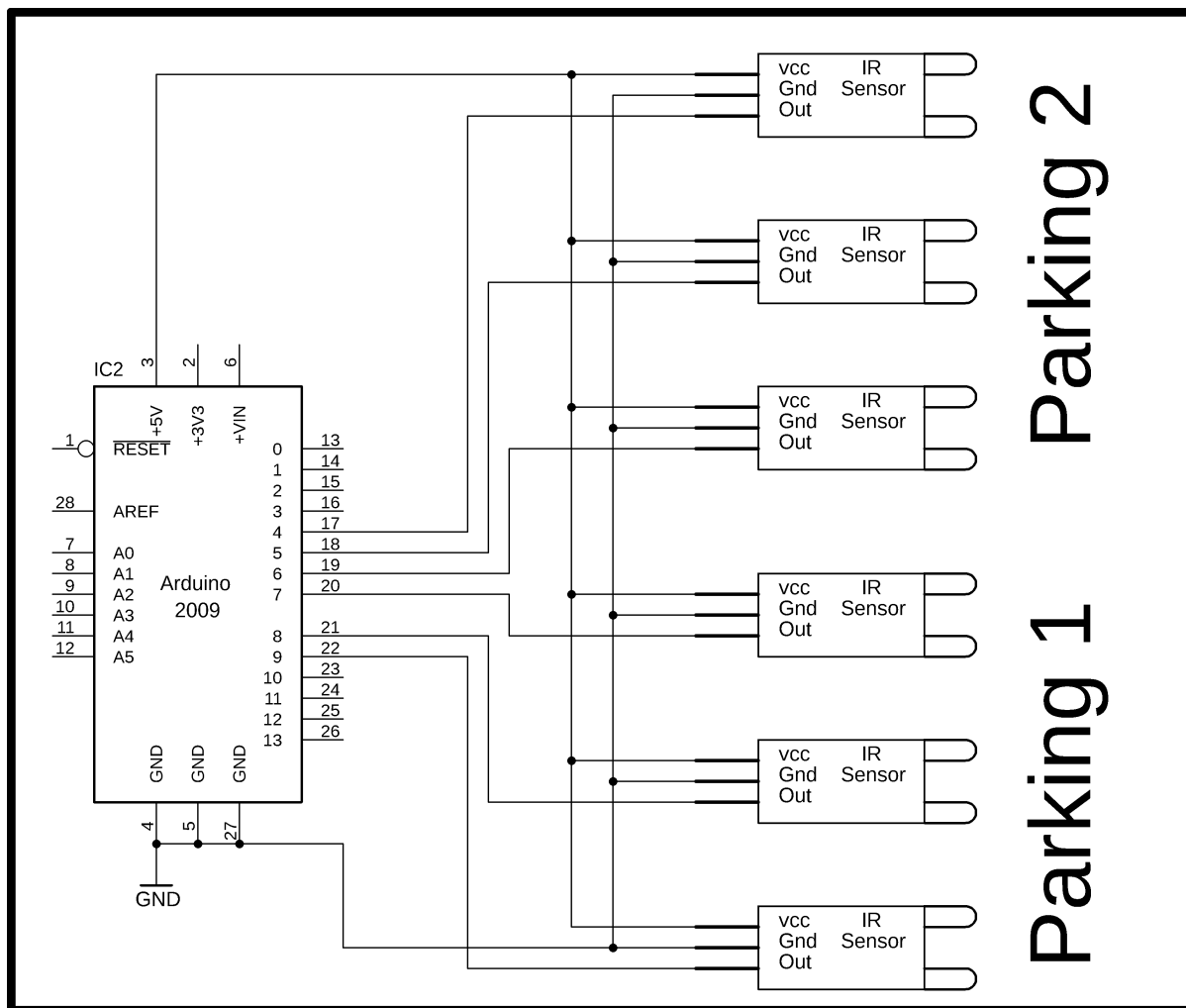


The Help section includes several documents that are easy to access, which comes along with the Arduino IDE. It consists of the number of options such as Getting Started, Environment, Troubleshooting, Reference, etc. We can also consider the image shown above, which includes all the options under the Help section.

Some documents like Getting started, Reference, etc., can be accessed without the internet connection as well. It will directly link us to the official website of Arduino.

## **System Architecture**

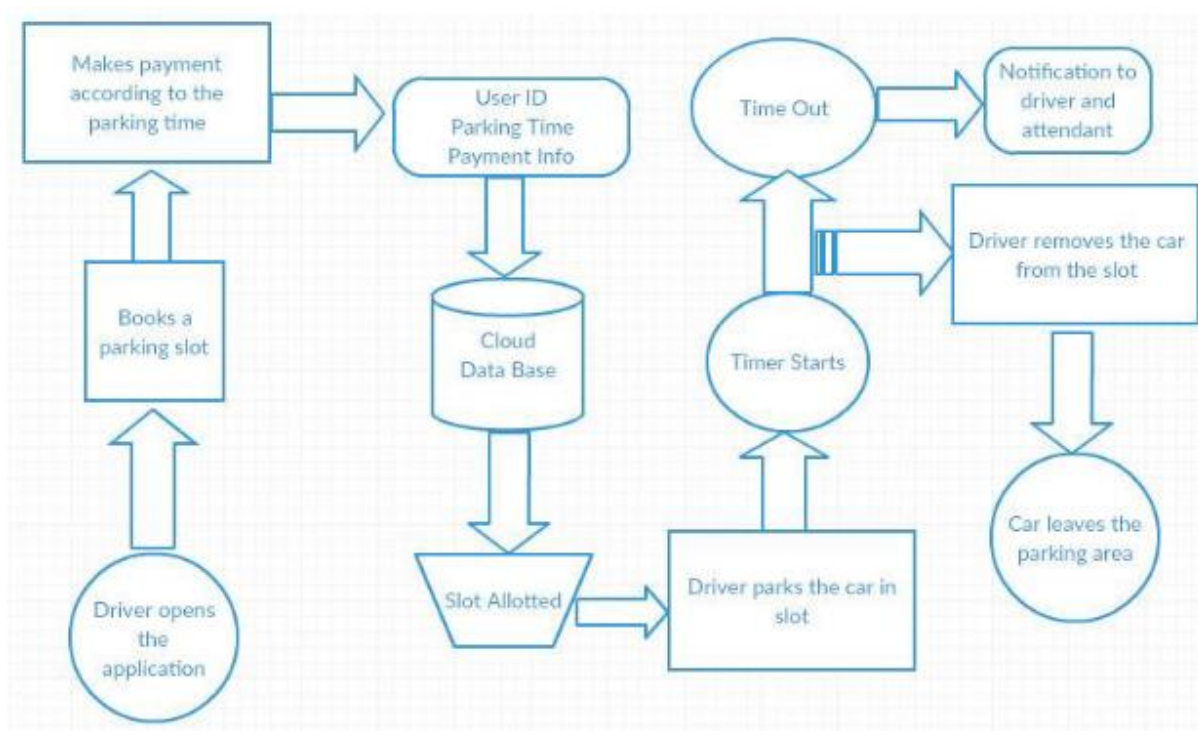
The below diagram shows the pin diagram of our model. It consists of one NODE MCU one dc motor, one 16\*2 LCD display and three IR sensors. The node mcu is the brain of our system which powers all the other devices. The 16\*2 LCD display is powered by node mcu by connecting jumper wires from the display to NODE MCU. The DC motor is also powered by node mcu with connecting its pins to node mcu. The IR sensor consists of three pins, where two pins refer to the power supply and ground and the other pins refer to the pin which is going to be connected in the NODE MCU. On successfully connecting all the components in the given figure now we have to connect the blink app. While using the blink app we have to specify the widgets used in our android app and the pin number to which they are connected to node Mcu in the actual model so that the mobile app will react exactly to the inputs provided in the model.



## CHAPTER 5

### IMPLEMENTATION

#### 5.1 Flowchart of the system



**Fig 5.1 Flow chart of the system**

Below are the steps that a driver needs to follow in order to park its car using our parking system.

- + **Step 1:** Install the smart parking application on your mobile device.
- + **Step 2:** On the 16\*2 display the number of vacant and filled spots are displayed so that the user can see the status of parking zone.
- + **Step 3:** Once the user logs into the app he would see the parking architecture with the Cars filled at which position and position which are empty.
- + **Step 4:** When the user is near to the parking IR detects sensors, he would receive a Notification on his app on which slot he can park his vehicle if there is an Empty slot.
- + **Step 5:** If there is no empty slot the user will be displayed with an appropriate Message on the mobile application.
- + **Step 6:** On the availability of parking area and user parking into the respective slot He/she would receive a message which states the start time of the parking and the slot in which he/she has parked.
- + **Step 7:** On successfully un-parking your vehicle from the parking slot the user will Receive a message which states the start time and end time of his parking Time and an amount which he needs to pay for the parking duration.

## 5.2 Design of the System

The picture shows the miniature model of the Automated Car Parking Lot.



Fig 5.2 Experimental Setup

This model has the capacity of containing two cars. There are two sensors at the entrance to detect the presence of a car before going inside or outside of the parking lot. The other two sensors are plotted inside the parking lot to detect the car individually for each parking slot. A DC Servo motor has been used at the entrance to open and close the gate according to the signals sent by the sensors through Arduino. The projection on the screen corresponds to the system model parking slots. This is a real time display regarding the status of the parking lot. As this is a web-based representation, anyone will be able to get the status of the parking lot by visiting the website on the URL through their cell phones, laptops, desktops and other internet supporting devices. The model of the parking lot has two parking slots. Thus, we can park a maximum number of two cars through the system.

We have used two IR sensors which when vehicle parked will show appropriate message to the user and when all the parking slots the dc motor would not open gate for the vehicle to be parked. Displaying of appropriate message for any action which takes place in the parking zone is done effectively and efficiently.

#### **Network Time protocol: -**

The Network Time Protocol is a networking protocol for clock synchronization between computer systems over packet-switched, variable-latency data networks. We have used NTP for fetching time from the NTP server so that we can show the start time and end time for the user when he parks or unparks his vehicle making information real-time.

#### **Blynk app: -**

Blynk app is a hardware-agnostic IoT platform with white label mobile apps, private cloud, device management, data analytics and machine learning. On using the blynk app we tried to pop notification to every possible event that is occurring in the parking zone.

Used a serial algorithm to display the slot number to the user who is going to park his vehicle. For example, we display the empty slot number in a serial manner which gets filled, if the slot 1 is filled and when another vehicle turns up, we display slot 2 and further like these for all other vehicles, and if any vehicle leaves the slot number, then we display the earliest slot number, not making the user to travel long if an initial spot is vacant.

## **CHAPTER 6**

### **RESULTS AND DISCUSSION**

#### **6.1 Initial Setup**

- ✚ The below diagram shows the initial case of the system when we turn on our project, which indicates the number of vacant and filled spots on a 16\*2 display LCD and similarly on the Blynk app.



**Fig 6.1** Shows both parking slot empty

## 6.2 Parking a vehicle

- The below diagram shows the status of the parking zone when a single vehicle is parked in the parking zone. Once when the user enters the parking detect sensor, he would receive a parking slot number on his mobile application which he is supposed to park his vehicle. Upon parking the vehicle in the respective slot and IR sensor successfully detecting the vehicle it would show a notification on the app the start time of the vehicle and the slot number in which the vehicle is parked and it would be similarly updated on the 16\*2 display.

### Stage 1:

When the car enters the parking area the IR sensor that is present before IN gate will detect the passing vehicle and the gate will be opened automatically.



Before reaching to IN gate



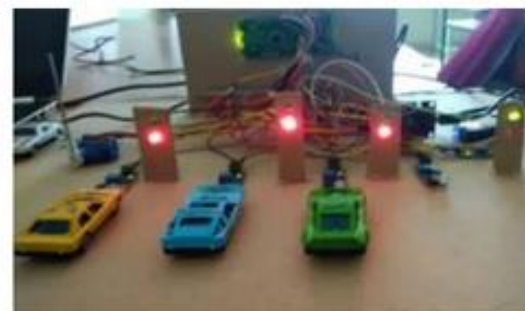
After reaching to IN gate

### Stage 2:

The car will enter into the parking area at the time person doesn't know which slot is empty, for this there will be an indication of LEDs for every slot when the Green Light glows the slot is empty when the red light glows the slot was filled. By the person easily know which slot is empty.



Before reaching to slot

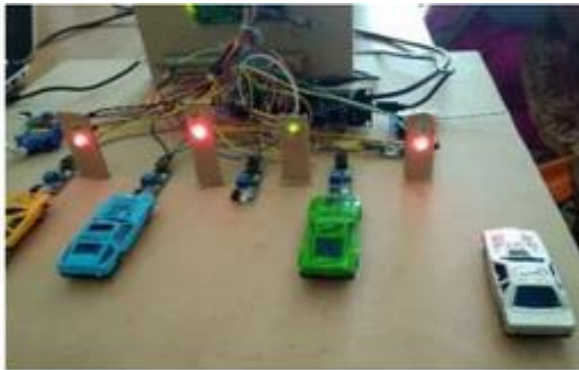


After reaching to slot



### ✚ Stage 3:

The operation of exit will be same as that of the entrance. When the car leaves the parking area, The IR sensor that is present before the OUT gate will detect the passing vehicle and the gate will be opened automatically.



Before reaching to OUT gate



After reaching to OUT gate

### ✚ Stage 4:

In front of the parking area, there will be an LCD display that is used to show the status of the parking slots, whether the parking is available or not.



## 6.3 Unparking a vehicle

- ✚ Unparking your vehicle from the parking slot would pop a notification on the application app stating the start time and the end time the user has parked his vehicle in the parking slot and an amount which the user needs to pay when he leaves the parking zone which is fixed for any duration.

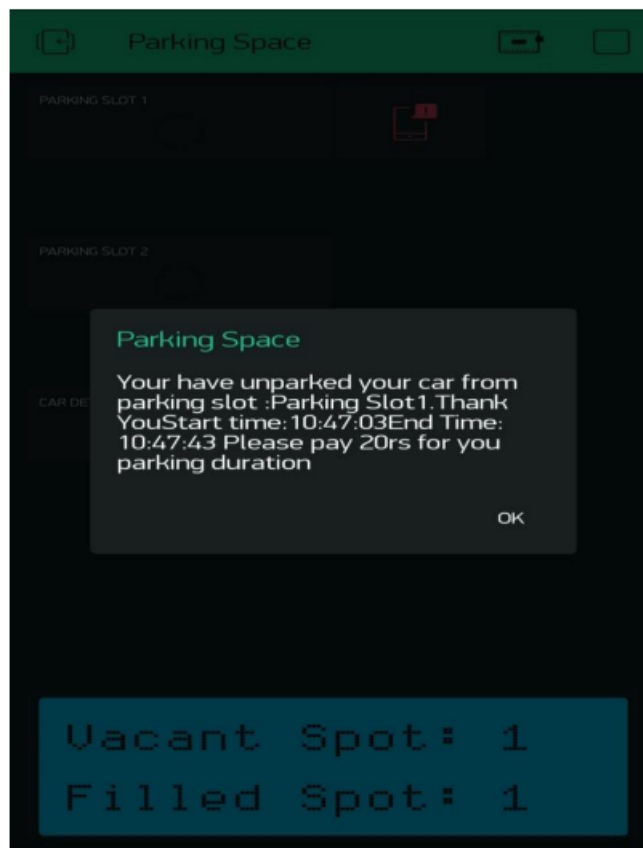
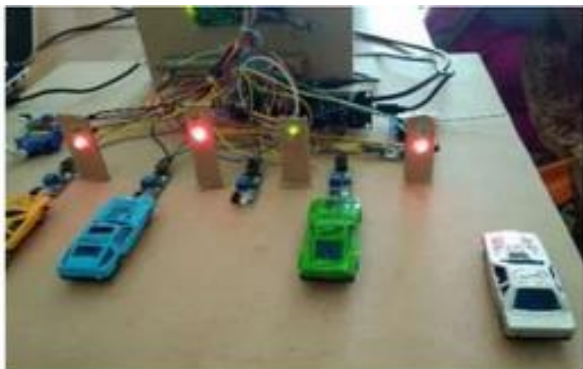


Fig 6.3 Unparking Vehicle



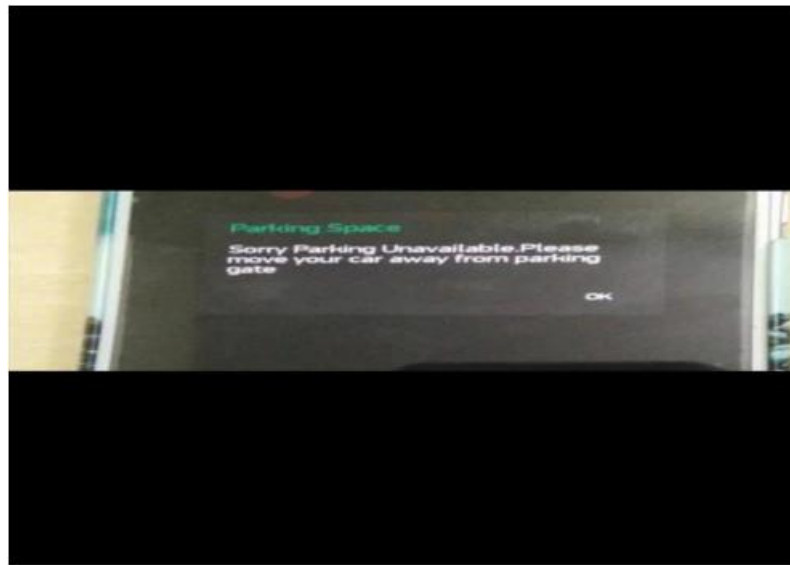
Before reaching to OUT gate



After reaching to OUT gate

#### 6.4 No available parking slots

- In a situation when all the slots are filled and a new vehicle comes near to the vehicle detecting sensor, the below message is popped on the user screen which displays that there are no parking slots available and the user can move his parking vehicle away from the parking zone.



**Fig 6.4** When there are no empty slots

## CHAPTER 7

### TESTING

Different cases have been explained and showed through the pictures in the following sections. All those two pictures correspond to each other while occurring at an event.

#### Case one

This case shows that all the parking slots are empty and therefore, the system will allow a car to enter into the parking zone. The 16\*2 LCD will display the number of vacant spot and filled spot and similarly it would be displayed on the application.



**Fig 7.1** Shows both parking slot empty on the model



Fig 7.2 App view of the parking system

## Case Two

This following case focuses on showing a slot number when the user is near to the parking detect sensor. It shows a parking slot number where the user should park his vehicle and upon parking it shows the start time of his parking.



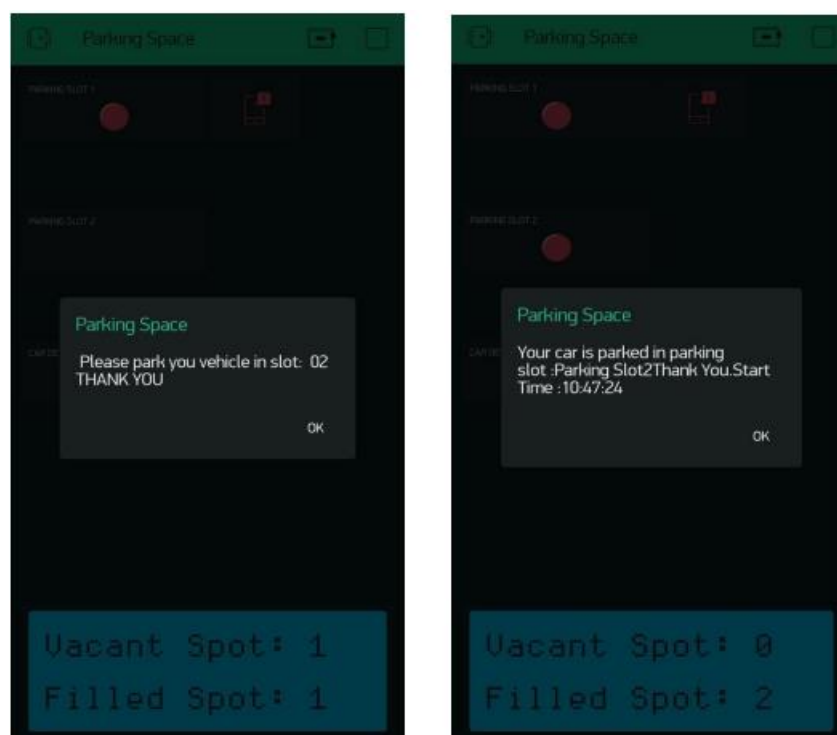
Fig 7.3 Case of one parking slot filled on the model



Fig 7.4 Shows the app view of vacant and filled spot

### Case Three

This following case shows the app view and the model view when one slot is filled and another vehicle turns up to the vehicle detect sensor. The DC motor then opens the gate for the user, and at that point of time the user receives a message specifying in which slot he can park his vehicle. On successfully parking his vehicle in the slot he would receive a notification stating his start time of parking and the respective slot number in which he has parked his vehicle.



## Case Four

This following case shows about unparking details where an user when unparks his vehicle from the parking slot he would be displayed with an appropriate message consisting of start time and end time of his parking. The user would be simultaneously allowed to pay a small amount which will be displayed on the application. On successfully unparking his vehicle the same would be updated in the count of 16\*2 display.



Fig 7.5 User unparking his vehicle from the slot



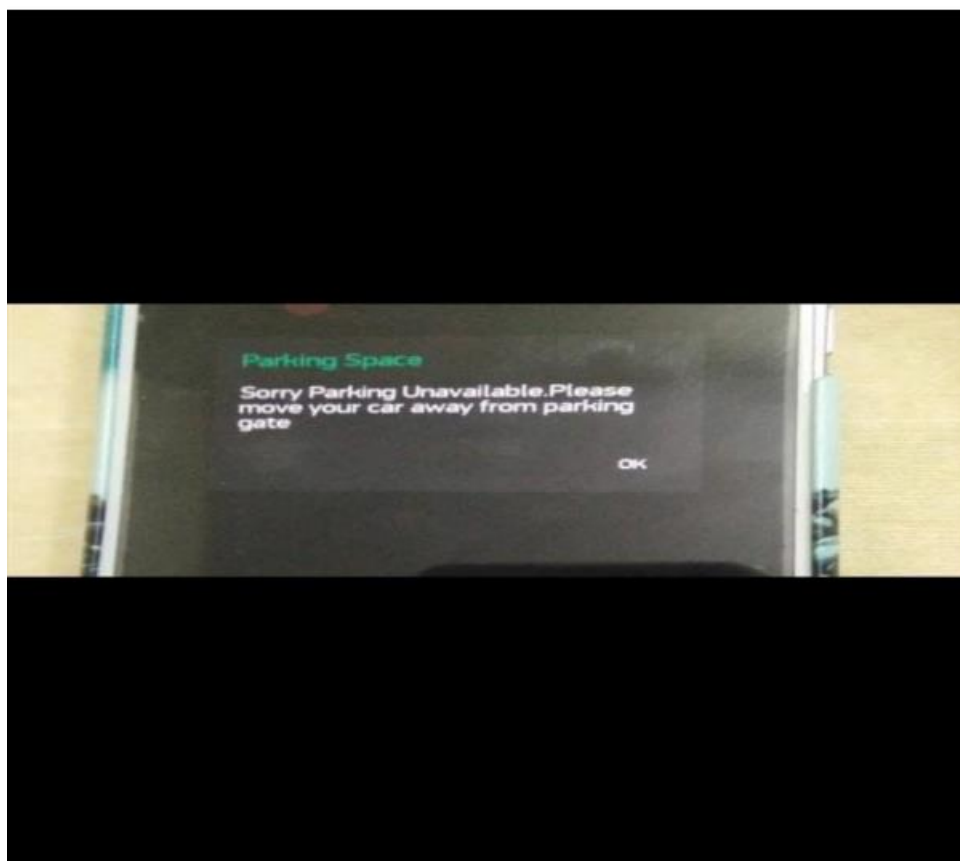
Fig 7.6 Application view of unparking .



**Fig 7.7** Updating the count of vacant and filled spot and displaying it .

### **Case Five**

In this case we would display a message stating that there are no more empty slots present in the parking zone. This message pop out when all the parking slots are filled and when a new vehicle turns up to the parking detect sensor requesting for an slot. Further the DC motor does not open the gates which makes it more sensible that there are no more parking slots available and the maintenance people should not upload any sign board indicating parking slots full. Saving human work in an efficient manner.



**Fig 7.8** Message on app indicating no parking slots available



**CHAPTER 8****CODE****#define BLYNK\_PRINT Serial****#include <ESP8266WiFi.h>****#include <BlynkSimpleEsp8266.h>****char auth[] = "TRZhGbb6BRTToVVw57N\_h8RDigtNf-O6X";****char ssid[] = "VITS";****char pass[] = "12345678";****WidgetLED led1(V1);****WidgetLED led2(V2);****WidgetLED led3(V3);****WidgetLED led4(V4);****WidgetLED led5(V5);****//#define IR D0****#define IR1 D1****#define IR2 D2****#define IR3 D3****#define IR4 D4****bool ir,ir1,ir2,ir3,ir4;****void setup() {****Serial.begin(15200);****pinMode(D0, INPUT);****//pinMode(IR1, INPUT);****//pinMode(IR2, INPUT);****//pinMode(IR3, INPUT);****//pinMode(IR4, INPUT);**

Blynk.begin(auth, ssid, pass);

}

void loop() {

ir = digitalRead(D0);

ir1 = digitalRead(IR1);

ir2 = digitalRead(IR2);

ir3 = digitalRead(IR3);

ir4 = digitalRead(IR4);

if(ir == 0)

{

led1.on();

}

else if (ir == 1)

{

led1.off();

}

if(ir1 == 0)

{

led2.on();

}

else if (ir1 == 1)

{

led2.off();

}

if(ir2 == 0)

{

```
led3.on();  
}  
else if (ir2 == 1)  
{  
  
led3.off();  
}  
  
if(ir3 == 0)  
{  
  
led4.on();  
}  
else if (ir3 == 1)  
{  
  
led4.off();  
}  
  
if(ir4 == 0)  
{  
  
led5.on();  
}  
else if (ir4 == 1)  
{  
  
led5.off();  
}  
Blynk.run();  
}
```

## CHAPTER 9

### CONCLUSION

The concept of Smart Cities has always been a dream for humanity. Since the past couple of years ago large advancements have been made in making smart cities a reality.

The growth of Internet of Things and Cloud technologies have given rise to new possibilities in terms of smart cities. Smart parking facilities and traffic management systems have always been at the core of constructing smart cities. In this project, we address the issue of parking and present an IoT based Cloud integrat

ed smart parking system. The system that we propose provides real time information regarding availability of parking slots in a parking area. Users from remote locations could book a parking slot for them by the use of our mobile application. The efforts made in this project are intended to improve the parking facilities of a city and thereby aiming to enhance the quality of life of its people.

#### Advantages of the smart parking system



- ✚ Optimized parking
- ✚ Reduced Traffic
- ✚ Reduced Pollution
- ✚ Enhanced User Experience
- ✚ Integrated Payments and POS
- ✚ Increased Safety
- ✚ Decreased Management Cost
- ✚ Shorter waiting time at the parking place
- ✚ Carbon Emission is reduced

#### Applications

- ✚ The smart parking system can be implemented in
- ✚ Shopping Malls
- ✚ Restaurants
- ✚ Theatres

#### 8.1 Future Work

- ✚ The future of the smart parking system is expected to be significantly influenced by the arrival of automated vehicles (AVs).
- ✚ Several cities around the world are already beginning to trial self-parking vehicles, specialized AV parking lots and robotics valets.
- ✚ This project can be enhanced for tracking vehicle speed on the road,
- ✚ Developing a smart parking solution within a city solves pollution problem.
- ✚ Addition of machine learning to store various other information of the Vehicle like its colour, design and number which would further add security.
- ✚ This project focuses on implementation of car parking place detection using Internet of Things.

-  The system benefits of smart parking go well beyond avoiding time wasting.
-  Developing a smart parking solution with in a city solves the pollution problem.

## REFERENCES

- [1] Abdul Ahad, Zishan Raza Khan, Syed Aqeel Ahmad, "Intelligent Parking System" Scientific Research Publishing, Vol.4, No.2, pp. 160-167, May 2016.
- [2] Dr Y Raghavender Rao," Automatic Smart Parking System using Internet of Things (IOT)" International Journal of Engineering Technology Science and Research, Vol.4, No.5, pp.225-258, May 2017
- [3] Suprit Atul Gandhi, Hasan Mohammad Shahid," Smart Parking System"
- [4] Asian Journal of Convergence in Technology, Vol.4, No.1, May 2017 Benson, J.P., T. O'Donovan, P. O'Sullivan, U. Roedig and C. Sreenan et al.," Car park management using wireless sensor networks", Proceedings of the 31st Conference on Local Computer Networks, Tampa, FL., USA., pp: 588-595 November 2006.
- [5] Geng Y. and Cassandras C. G, "A new smart parking system based on optimal resource allocation and reservations," in Proc. IEEE Conf. Intell. Transp. Syst. pp. 979– 984, July 2011.
- [6] M. M. Rashid, A. Musa, M. Ataur Rahman, and N. Farahana, A. Farhana, "Automatic Parking Management System and Parking Fee Collection Based on Number Plate Recognition.", International Journal of Machine Learning and Computing, Vol. 2, No. 2, April 2012, Published 2014.
- [7] Arduino.cc. (2018). Arduino - ArduinoMega2560. , retrieved date: 21Oct.2018, online available at:<https://www.arduino.cc/en/Guide/ArduinoMega2560>