**ABOUT**

People no longer rely on public transportation as a result of the rapid urbanisation. They move around in their cars. As a result, traffic grows. The hardest issue for travellers in a metropolis is finding a place to park. Due to traffic congestion, it not only results in increased time and fuel loss for other cars as opposed to only the drivers seeking for parking. There are now more people driving, which has made parking and traffic problems worse. Increasing staff is now the most often utilised technique for dealing with such volume. The parking of automobiles has become a problem, even at shopping malls, trade centres, and business parks. We have all encountered the commotion, confusion, and lengthy lines necessary to locate a suitable parking space in such locations. There is currently no mechanism in place to obtain information about parking availability online, making it exceedingly difficult to find parking in congested places. Imagine not having to walk around to verify the availability of parking spaces since you could obtain information about it on your phone. The Internet of Things-based smart parking system can address this issue. You may quickly access the parking space availability online by using the IoT-based parking system. The parking system may be fully automated using this technique. Entry, payment, and exit may all be completed automatically. Therefore, we're employing NodeMCU, five IR sensors, and two servo motors to construct an Internet of Things-based car parking system. At the entry and exit gates, two IR sensors are used to detect the automobile, and three IR sensors are utilised to determine whether a parking space is available. In accordance with the sensor value, servo motors are employed to open and close the gates. Here, we demonstrate how to publish data in the cloud that can be seen from any location using the Adafruit IO platform. One of the most popular and rapidly expanding Smart City solutions worldwide is smart parking. A few organisations that have started to appreciate the substantial advantages of automated parking technologies include airports, colleges, retail malls, and civic garages. Smart parking is made feasible by the capacity to connect, analyse, and automate data obtained from devices that are supported by and referred to as the Internet of Things. Utilizing low-cost sensors, real-time data, and applications that let users monitor available and unavailable parking spaces are all part of smart parking. The idea is to automate the process and cut down on the amount of time spent manually looking for the best parking level, space, or even lot. Some systems will provide a full range of services, including online payments, alerts of remaining parking time, and even car-searching capabilities for particularly big lots. The user and lot owner can both gain a lot from a parking solution. Any local authority or business would make a wise investment by implementing a smart parking solution. It is crucial to develop a well-planned and convenience-driven parking solution that can be used internationally as the world's population continues to increase and urbanise. Work costs and resource depletion are reduced by increased automation and decreased manual labour. The user's perception of a business or commercial entity's brand might soar thanks to a smooth experience. Visitors will undoubtedly be astonished by the state-of-the-art technology and convenience features, whether the location is a retail store, an airport, or a corporate business centre. Briefly stated, Smart Parking is a parking solution that may comprise counting sensors or in-ground Smart Parking sensors. In order to determine if parking spaces are empty or occupied, these devices are typically placed adjacent to or implanted into parking spaces. Real-time data collecting enables this. The information is subsequently sent to a smart parking mobile application or website, which informs its users of the availability.

**Parts used with their technical specifications**

*A. Hardware*

1. NodeMCU ESP8266
2. Jumpers
3. Servo Motor
4. Power Supply
5. IR Proximity Sensor

*B. Software Requirement*

1. Adafruit IO (Online Service)
2. Arduino IDE

**SPECIFICATIONS**

1. **NODEMCU ESP8266**



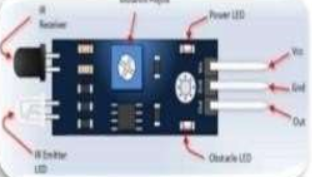
1. NodeMCU ESP8266 is an open-source Lua-based firmware and development board specially targeted for IoTbased applications.
2. It includes firmware that runs on the ESP8266 WiFi SoC from Express if Systems and hardware which is based on the ESP-12 module, and like this, it can also be programmed using Arduino IDE and can act as both Wi-Fi Hotspot or can connect to one.
3. It has one Analog Input Pin, 16 Digital I/O pins along with the capability to connect with serial communication protocols like SPI, UART, and I2C.
4. NodeMCU has 128 KB RAM and 4MB of Flash memory to store data and programs
5. **Servo Motor**



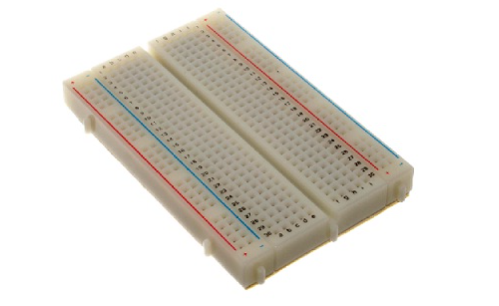
1. Operating Voltage is +5V typically
2. Torque: 2.5kg/cm
3. Operating speed: 0.1s/60°
4. Gear Type: Plastic
5. Rotation: 0°-180°
6. Weight of motor: 9gm
7. **Jumper Wire**



1. Standard 0.1" (2.54mm) spacing when placed next to each other
2. Length: 200mm (7.87")
3. Wire Colors: brown, red, orange, yellow, green, blue, purple, grey, white, black (Each cable includes 4 of each color)
4. Fits breadboard
5. Weight: 31g
6. **IR Proximity Sensor**

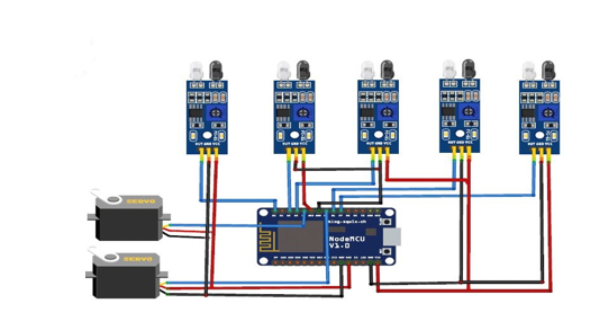


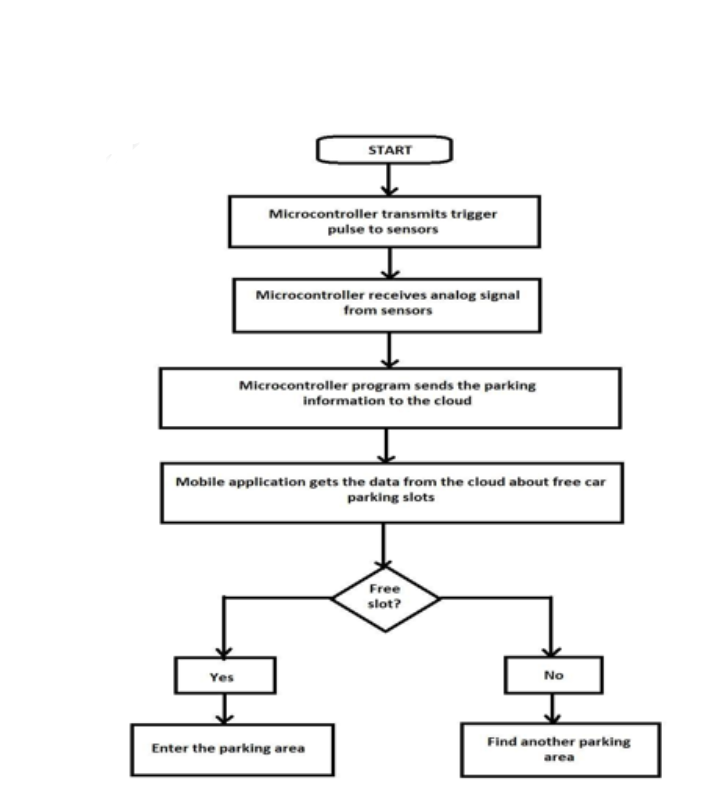
1. IR transmitter
2. Pin easy interface connectors
3. Indicator LED & Power LED
4. Distance 2cm to 30cm
5. Active Low on object detection
6. 3.3 to 5V operation
7. **BREADBOARD**

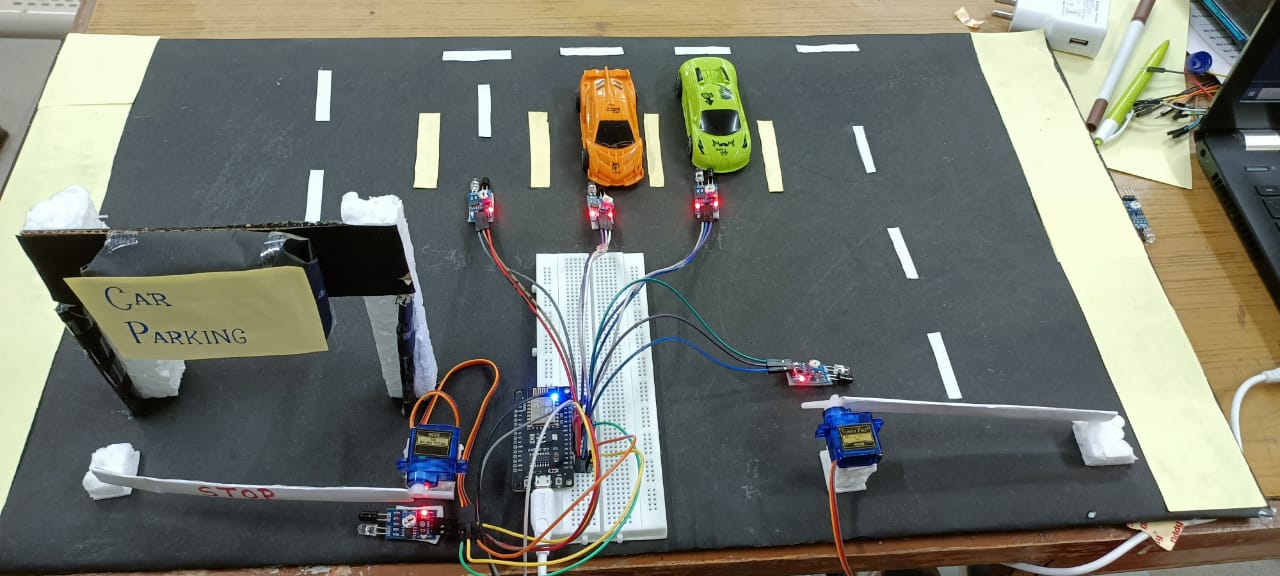
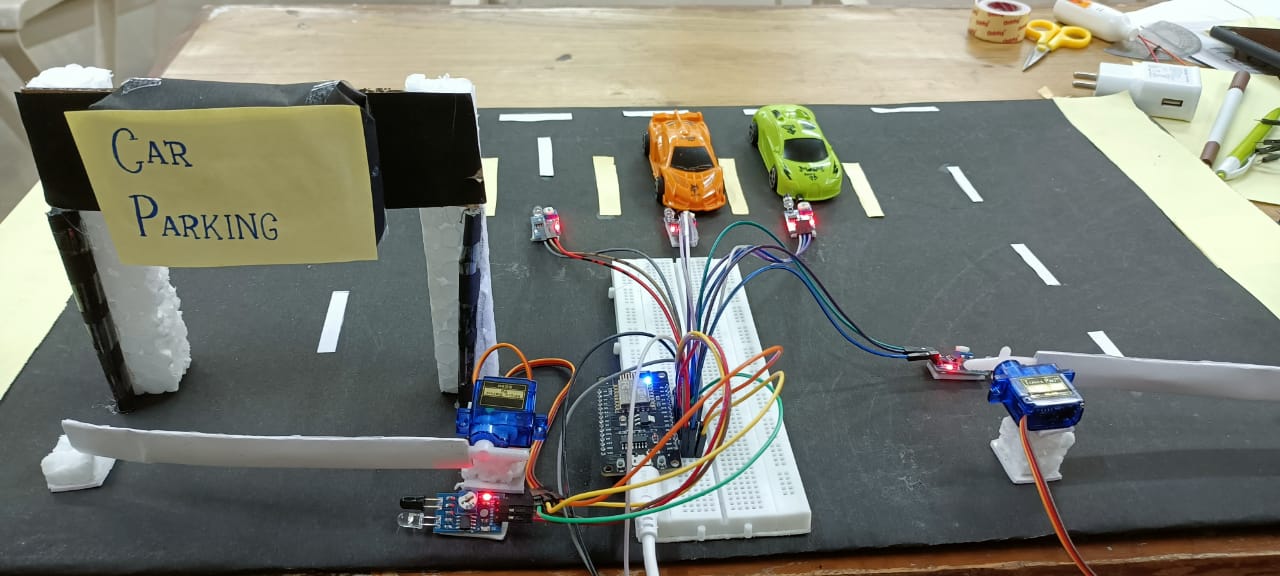
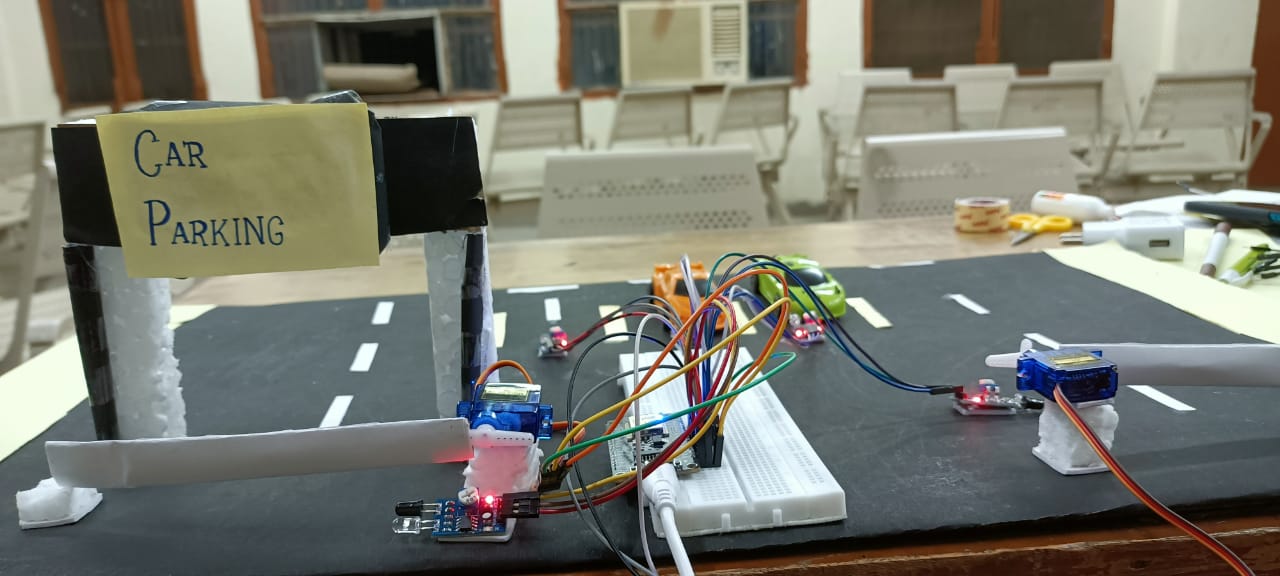
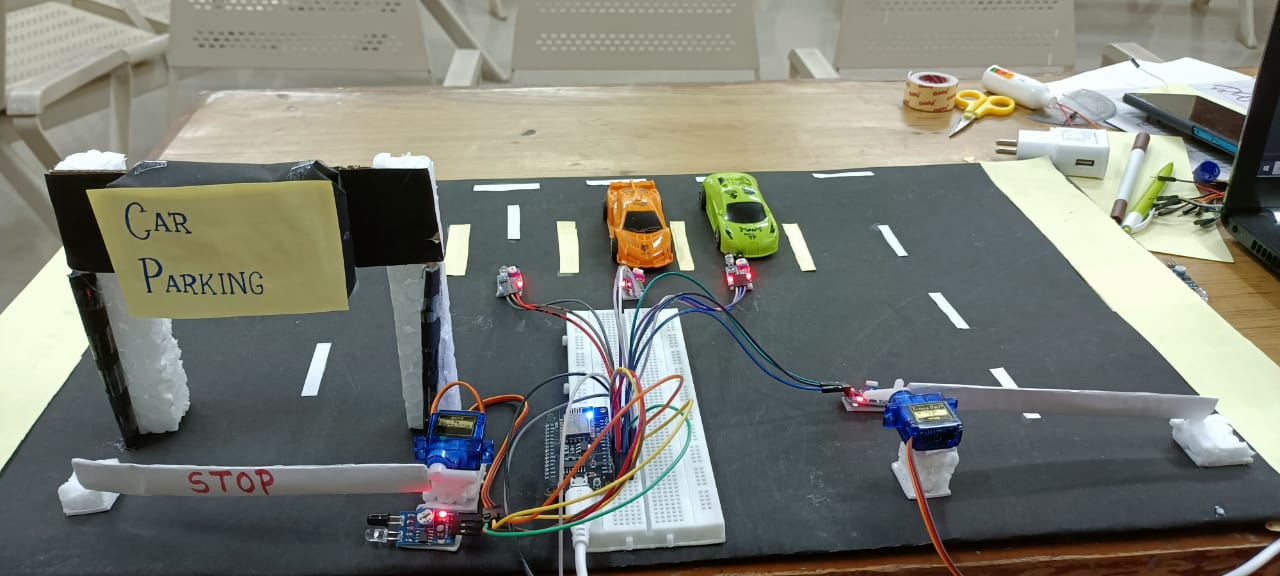
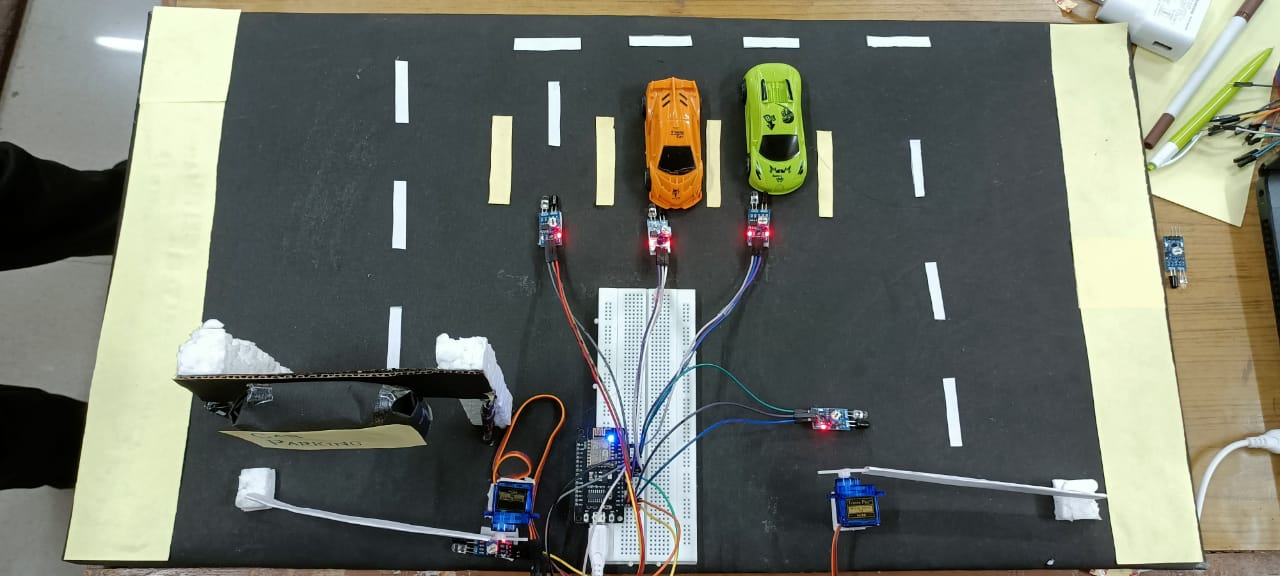


1. Distribution Strips are two
2. Wire Size is 21 to 26 AWG wire
3. Tie Points are two hundred
4. Withstanding Voltage is 1,000V AC
5. Tie points within IC are 630
6. Insulation Resistance is DC500V or 500MΩ
7. Dimension is 6.5\*4.4\*0.3 inch
8. Rating is 5Amps
9. ABS plastic through color legend
10. ABS heat Distortion Temperature is 183° F (84° C)Hole or Pitch Style is 2.54mm.

**Circuit diagram**





**Actual pics of the project**

**Construction**

In this Smart Parking System using IOT, we are using five IR Sensors and two servo motors. IR sensors and Servo motors are connected to the NodeMCU. NodeMCU controls the complete process and sends the parking availability and parking time information to Adafruit IO so that it can be monitored from anywhere in the world using this platform. Two IR sensors are used at entry and exit gate so that it can detect the cars at entry and exit gate and automatically open and close the gate.

Two servo motors are used as entry and exit gate, so whenever the IR sensor detects a car, the servo motor automatically rotates from 45° to 140°, and after a delay, it will return to its initial position. Another three IR sensors are used to detect if the parking slot is available or occupied and send the data to NodeMCU. Adafruit IO dashboard also has two buttons to manually operate the entry and exit gate.

**SOFTWARE INSTALLATION**

1. To use Adafruit IO, first, you have to create an account on Adafruit IO. To do this, go to the [Adafruit IO](https://io.adafruit.com/) website and click on ‘**Get started for Free’** on the top right of the screen.
2. After finishing the account creation process, log in to your account and click on ‘**AIO Key’** on the top right corner to get your account username and AIO key.
3. Now, after this, you need to create a feed. To create a feed, click on ‘**Feed**.’ Then click on ‘**Actions,**’ and then on ‘**Create a New Feed’** as shown in the image below
4. After this, a new window will open to enter the Name and Description of the feed. The writing description is optional.
5. Click on ‘Create,’ after this; you will be redirected to your newly created feed. For this project, we created a total of **nine feeds for exit gate, entry gate, slot 1 entry & exit, slot 2 entry & exit, and slot 3 entry & exit**. After creating feeds, now create an Adafruit IO dashboard to show all of these feeds on a single page. **To create a dashboard**, click on the Dashboard option and then click on the ‘**Action**,’ and after this, click on ‘**Create a New Dashboard.**
6. As the dashboard is created now, we will add our feeds to the dashboard. **To add a feed**, click on the ‘+’ in the top right corner.
7. First, we will add two RESET buttons blocks for Entry and Exit gate and then seven TEXT blocks for parking details. To add a button on the dashboard click on the RESET block
8. To create the rest of the blocks follow the same procedure, but instead of creating a RESET block, create a TEXT block so that you can show the parking details. After creating all the blocks, my dashboard looks like below. You can edit the dashboard by clicking on the settings buttons